VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

ROZPOZNÁNÍ ZVUKOVÝCH UDÁLOSTÍ POMOCÍ HLUBOKÉHO UČENÍ
DEEP LEARNING BASED SOUND EVENT RECOGNITION

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Jakub Bajzík**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Jiří Přinosil, Ph.D.**
SUPERVISOR

BRNO 2019

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Audio inženýrství**
Ústav telekomunikací

Student: Bc. Jakub Bajzík ***ID:*** 174264
Ročník: 2 ***Akademický rok:*** 2018/19

NÁZEV TÉMATU:

Rozpoznání zvukových událostí pomocí hlubokého učení

POKYNY PRO VYPRACOVÁNÍ:

Nastudujte teoretické možnosti využití technik hlubokého učení pro rozpoznání vytipovaných zvukových událostí. Na základě získaných poznatků proveďte návrh, implementaci a ověření algoritmu pro klasifikaci zdrojů vybraných zvukových událostí. Součástí práce bude rovněž příprava rozsáhlé databáze zvukových událostí, která bude vhodná pro trénování a testování navrženého algoritmu.

DOPORUČENÁ LITERATURA:

[1] FOGGIA, Pasquale, et al. Reliable detection of audio events in highly noisy environments. Pattern Recognition Letters, 2015, 65: 22-28.

[2] JIA, Yangqing, et al. Caffe: Convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM international conference on Multimedia. ACM, 2014. 675-678.

Termín zadání: 1.2.2019 ***Termín odevzdání:*** 16.5.2019

Vedoucí práce: Ing. Jiří Přinosil, Ph.D.
Konzultant:

prof. Ing. Jiří Mišurec, CSc.
předseda oborové rady

ABSTRAKT

Táto práca sa zaoberá problematikou spracovania a rozpoznávania udalostí v zvukovom signále. Práca skúma možnosť využitia vizualizácie zvukového signálu a následné použitie konvolučných neurónových sietí ako klasifikátoru pre rozpoznanie v reálnom použití. Vybrané zvukové udalosti sú výstrely zo zbraní umiestnené do zvukového pozadia ako je ruch ulice, ľudský hlas, zvuky zvierat a iné formy náhodného šumového pozadia. Pred samotnou implementáciou je vytvorená rozsiahla databáza s rôznymi parametrami výstrelov najmä charakteru dozvuku a časovej polohy v rámci spracovávaného úseku. V práci sú použité voľne dostupné platformy Keras a TensorFlow pre prácu s neurónovými sieťami.

KĽÚČOVÉ SLOVÁ

Rozpoznávanie zvukových udalostí, strojové učenie, neurónová sieť, spracovanie signálu

ABSTRACT

This paper deals with processing and recognition of events in audio signal. The work explores the possibility of using audio signal visualization and subsequent use of convolutional neural networks as a classifier for recognition in real use. Recognized audio events are gunshots placed in a sound background such as street noise, human voice, animal sounds, and other forms of random noise. Before the implementation, a large database with various parameters, especially reverberation and time positioning within the processed section, is created. In this work are used freely available platforms Keras and TensorFlow for work with neural networks.

KEYWORDS

Sound recognition, machine learning, neural network, signal processing

BAJZÍK, Jakub *Rozpoznání zvukových událostí pomocí hlubokého učení*: diplomová práca. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2019. 64 s. Vedúci práce bol Ing. Jiří Přinosil, Ph.D.

VYHLÁSENIE

Vyhlasujem, že som svoju diplomovoú prácu na tému „Rozpoznání zvukových událostí pomocí hlubokého učení“ vypracoval(a) samostatne pod vedením vedúceho diplomovej práce, využitím odbornej literatúry a ďalších informačných zdrojov, ktoré sú všetky citované v práci a uvedené v zozname literatúry na konci práce.

Ako autor(ka) uvedenej diplomovej práce ďalej vyhlasujem, že v súvislosti s vytvorením tejto diplomovej práce som neporušil(a) autorské práva tretích osôb, najmä som nezasiahol(-la) nedovoleným spôsobom do cudzích autorských práv osobnostných a/alebo majetkových a som si plne vedomý(-á) následkov porušenia ustanovenia § 11 a nasledujúcich autorského zákona Českej republiky č. 121/2000 Sb., o práve autorskom, o právach súvisiacich s právom autorským a o zmene niektorých zákonov (autorský zákon), v znení neskorších predpisov, vrátane možných trestnoprávnych dôsledkov vyplývajúcich z ustanovenia časti druhej, hlavy VI. diel 4 Trestného zákoníka Českej republiky č. 40/2009 Sb.

Brno ....

podpis autora(-ky)

POĎAKOVANIE

Rád by som poďakoval vedúcemu diplomovej práce pánovi Ing. Jiřímu Přinosilovi, Ph.D. za odborné vedenie, konzultácie, trpezlivosť a podnetné návrhy k práci.

Brno . . . . . .

. . . . . .

podpis autora(-ky)

OBSAH

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

ÚVOD

Rozpoznávanie objektov a udalostí pomocou hlbokého učenia bolo donedávna spájané predovšetkým s obrazovým signálom. V prípade spracovania zvukových signálov mimo hudobných sú dnes dobre známe najmä metódy spracovania ľudského hlasu. Tieto metódy sú vyvíjané za účelom zníženia dátových tokov v telekomunikačných prostriedkoch, rozpoznávanie hovorcu alebo prevodu písaného textu na syntetickú reč. V každodennom živote sa s týmito aplikáciami stretáme v chytrých telefónoch, na webových stránkach, v systémoch inteligentných domácností a v iných oblastiach IoT.

V období posledných rokov sa však častejšie skloňuje použitie známych postupov na rozpoznanie okolitých zvukových udalostí prostredia, ktoré môžu byť výbuch, výstrel zo zbrane, siréna, poplašné zariadenie auta, detský plač, rozbitie okna a iné udalosti spájané s potenciálnym nebezpečenstvom. Využitie takto naučených algoritmov je najmä zvýšenie bezpečnosti majetku a osôb. Implementácia je možná v domových alebo priemyselných systémoch ochrany, v automobiloch pre upozornenie nepočujúcich alebo slabo počujúcich vodičov pred sirénou, v domácnostiach na upozornenie rodiča na plačúce dieťa a v širokom spektre pomocných zariadení najmä pre nepočujúce osoby.

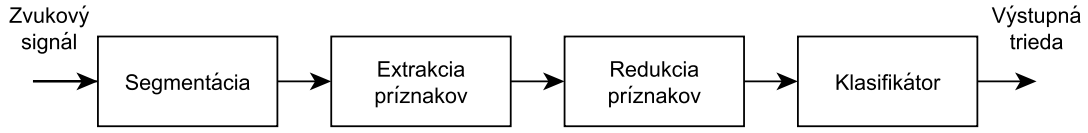
Obsah tejto práce je zameraný na možnosť využitia obrazovej reprezentácie zvukového signálu ako príznaku pre učenie neurónovej siete. Okrem často používaného spektrogramu sú hľadané nové reprezentácie dostatočne vypovedajúce o charaktere danej zvukovej udalosti. Cieľom práce je nájsť najvhodnejšiu, prípadne kombináciu viacerých reprezentácií zvukového signálu pre učenie algoritmov vyvinutých na rozpoznanie obrazových dát. Najvhodnejší postup by nemal dosahovať len najvyššiu presnosť rozpoznania udalostí ale tiež by mal byť použiteľný v prípade prekrývania viacerých zvukových udalostí a vo vysoko rušivom prostredí.

1 KLASIFIKÁCIA ZVUKOVÝCH UDALOSTÍ

1.1 Obecný návrh algoritmu

1.1.1 Tradičný postup

Obecne pri spracovaní zvukového signálu akéhokoľvek charakteru môžeme postup rozdeliť do základných krokov podľa blokového diagramu na obrázku 1.1. V prvom kroku je audio signál segmentovaný na kratšie úseky z ktorých je následne extrahovaný vektor príznakov. Príznakmi označujeme parametre alebo atribúty vypočítané zo zvukového signálu, ktoré môžu byť spektrálne koeficienty, melovské kepstrálne koeficienty, lineárne predikčné koeficienty a pod. V prípade veľkého množstva príznakov je vhodné vybrať tie najrelevantnejšie a najdôležitejšie. Redukcia môže byť prevedená výberom príznakov metódami ako minimálna redundancia a maximálna relevancia alebo transformáciou príznakov algoritmami ako analýza hlavných komponentov. Posledným krokom je určenie pravdepodobnosti zaradenia neznámeho zvuku do konkrétnej triedy pomocou štatistického klasifikátora, ktorým môže byť umelá neurónová sieť [1].



Obr. 1.1: Blokový diagram postupu rozpoznania zvukovej udalosti.

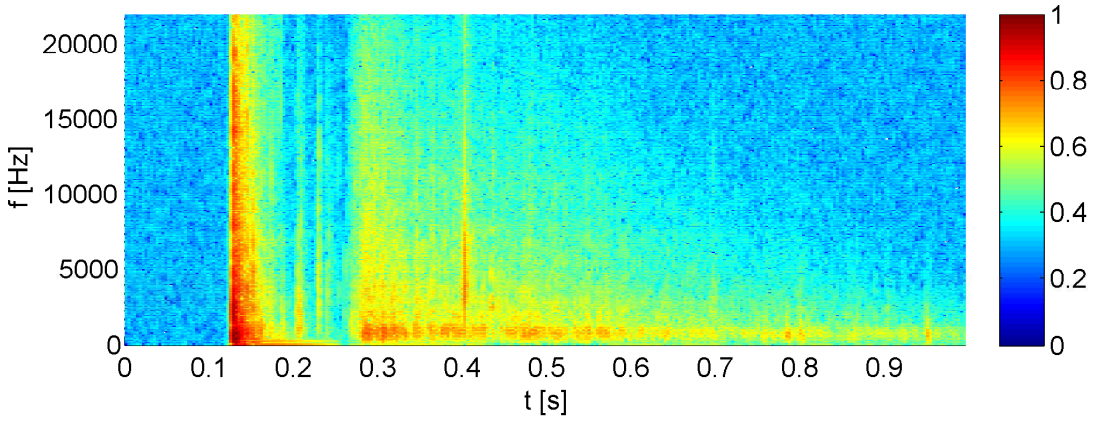
1.1.2 Navrhnutý postup

Úlohou algoritmu navrhnutého v tejto práci je rozpoznať zvuk výstrelu od náhodného pozadia. Jedná sa teda o binárnu klasifikáciu. Navrhnutý postup rozpoznania principiálne vychádza z tradičného postupu, no namiesto vektoru príznakov sú počítané dvojrozmerné matice skladané za seba. Zmenou teda je, že príznakový priestor je trojdimenzionálny, preto je nutné pred prepojením s neurónovou sieťou matice príznakov previesť na jednorozmerný vektor. Na to slúži konvolučný model siete. Výhodou tohto postupu je, že pri prekrytí jednotlivých matíc existuje istá priestorová súvislosť medzi príznakmi. Dnes je známych veľa postupov dvojdimenzionálnej vizualizácie zvukového signálu, z ktorých niekoľko vybraných bude popísaných v následnej sekcií.

1.2 Vizualizácia zvuku

1.2.1 Spektrogram

Základnou a najpoužívanejšou grafickou reprezentáciou zvukového signálu je spektrogram. Zobrazuje vývoj frekvenčného spektra v čase, pričom modul spektra je zobrazený vo farebnej škále. Pre prevod signálu z časovej do frekvenčnej oblasti sa používa krátkodobá fourierová transformácia.



Obr. 1.2: Reálny spektrogram zvuku výstrelu zo zbrane.

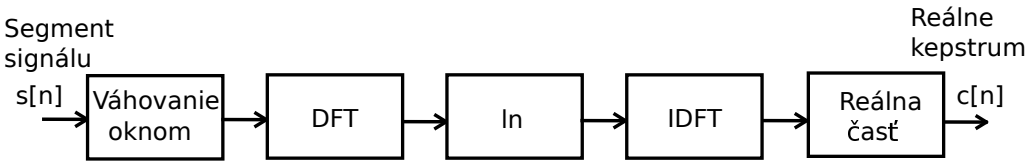
Signál je najskôr rozdelený do časových segmentov a následne váhovaný určitým typom okna. Váhové okno vyhladzuje ostré prechody medzi segmentami a zároveň znižuje mieru presakovania susedných frekvencií v spektre signálu. Často sa používa 50% prekrytie segmentov. V ideálnom prípade predpokladáme, že na tomto úseku nedochádza k príliš veľkým zmenám parametrov signálu, preto voľba dĺžky segmentov závisí najmä od povahy signálu. Ak však zvolíme dĺžku časového segmentu príliš krátku, výrazne zmenšíme rozlíšenie vo frekvenčnej oblasti. Heisenbergov princíp neurčitosti hovorí, že presnosť určenia istých dvojíc veličín je obmedzená, teda nieje možné dosiahnuť maximálne rozlíšenie oboch veličín zároveň [2]. Pre časové a frekvenčné rozlíšenie platí obmedzenie podľa vzťahu 1.1.

$$\Delta_t \Delta_f \geq \frac{1}{4\pi} \tag{1.1}$$

Rozlíšenie výsledného spektrogramu ako digitálneho obrazu je vhodné zjednotiť pre celú množinu trénovacich dát. Podľa publikácie [8] dosahuje samotný spektrogram pri rozpoznávaní enviromentálnych zvukov použitím neurónových sietí najlepšie výsledky, preto bude použitý v tejto práci v praktickej časti.

1.2.2 Reálny kepstrogram

Na rozdiel od spektrogramu je kepstrogram pre ľudskú intuíciu pomerne imaginárny pojem. Postup kepstrálnej analýzy je zobrazený v blokoch na obrázku 1.3. Rovnako ako pri frekvenčnej analýze je signál najskôr rozdelený do segmentov, váhovaný oknom a transformovaný do spektrálnej oblasti pomocou diskrétnej fourierovej transformácie DFT. Reálne výkonové kepstrum následne získame prirodzeným logaritmovaním spektrálnach koeficientov a spätnou DFT.



Obr. 1.3: Bloková schéma kepstrálnej analýzy diskrétneho signálu [1].

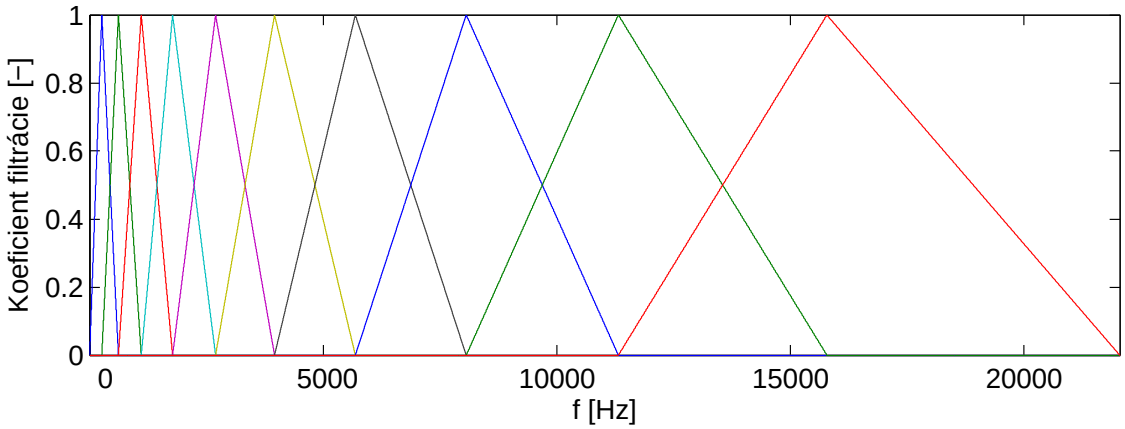
V tejto oblasti hovoríme o reálnych kepstrálnych koeficientoch a quefrencií. Tieto pomenovania vznikli ako anagramy od slov spektrum a frekvencia. Z tvaru reálneho kepstra hlások je možné zistiť parametre hlasového traktu, preto je táto analýza využívaná najmä v oblasti spracovania reči. Časový vývoj kepstrálnych koeficientov v jednotlivých segmentoch predstavuje reálny kepstrogram.

1.2.3 Melovské kepstrálne koeficienty

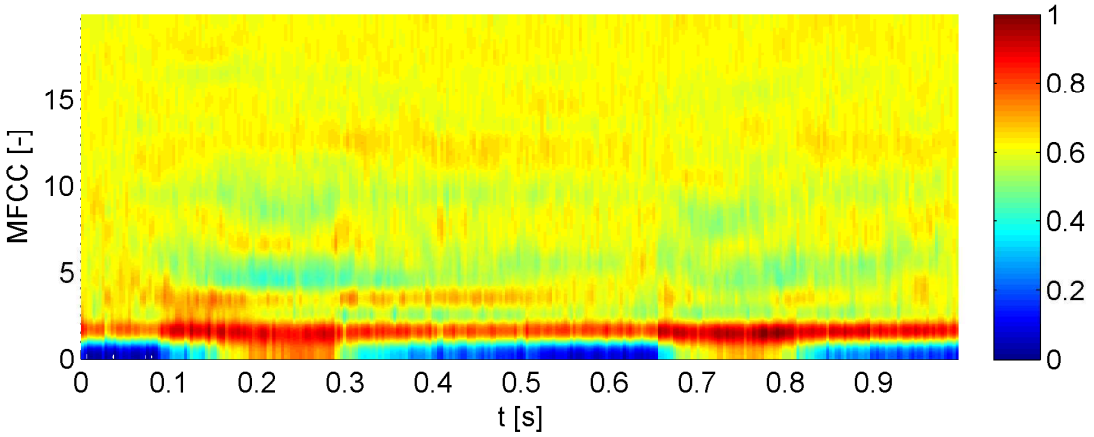
Melovské kepstrálne koeficienty MFCC úzko súvisia s kepstrálnou analýzou a vychádzajú z nelineárnych a maskovacích vlastností ľudského sluchu. Keďže ľudské ucho neregistruje zmenu frekvencie lineárne, ku každej z meraných frekvencií môžeme priradiť subjektívnu výšku tónu v jednotkách mel podľa vzťahu 1.2.

$$mel(f) = 2595 \log_{10}(1 + \frac{f}{700}) \tag{1.2}$$

Prepočtom lineárne rozmiestnených trojuholníkových filtrov v melovskej škále do frekvenčnej dostaneme nelineárne rozloženú banku filtrov, z ktorej následne počítame energiu v každom pásme. Prirodzeným logaritmovaním získame melovské spektrálne koeficienty a po prevedení spätnej diskrétnej kosínovej transformácie koeficienty MFCC [1]. Podobne ako v predchádzajúcich prípadoch získame dvojrozmerný obraz postupným skladaním koeficientov v jednotlivých časových segmentoch a ich farebným škálovaním.



Obr. 1.4: Banka desiatich filtrov pri vzorkovacej frekvencií $f_s = 44,1\,kHz$.



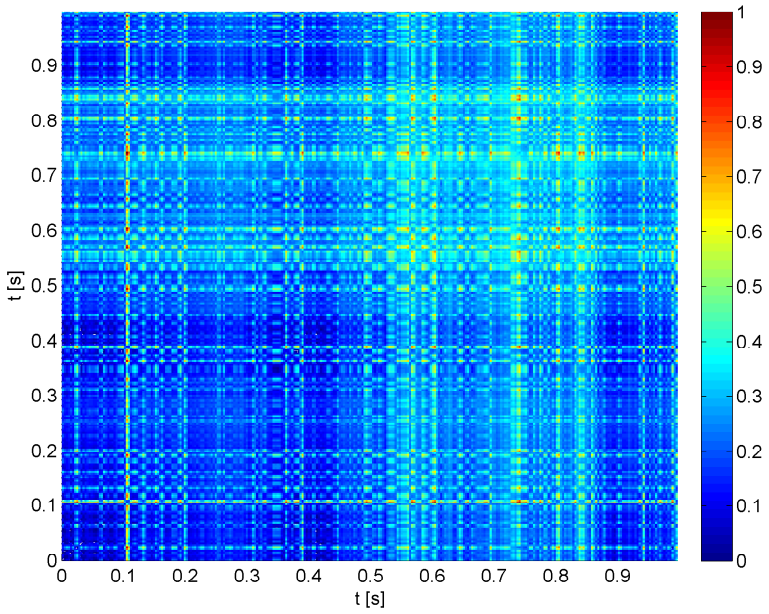
Obr. 1.5: Časová zmena MFCC koeficientov nahrávky štekotu psa.

1.2.4 Miera podobnosti

Pre vzorkovaný zvukový signál a jeho vlastnosti je možné zistiť mieru vlastnej podobnosti založenú na vzdialenostiach. Táto technika sa používa na analýzu globálnej štruktúry hudobných diel [4]. Pre zobrazenie podobnosti použijeme maticu vlastnej podobnosti $\boldsymbol{S}$ s rozmermi N×N. Pre jednotlivé prvky matice platí vzťah 1.3, kde $d(v_i, v_j)$ je funkcia vyjadrujúca podobnosť dvoch vektorov vlastností signálu. Medzi najčastejšie spôsoby výpočtu podobnosti patrí absolútny rozdiel a eukleidova miera.

$$\boldsymbol{S}(i,j) = d(v_i, v_j) \quad i,j = 1, ..., N \tag{1.3}$$

Vertikálna a horizontálna os zobrazenia predstavuje časovú postupnosť. Najväčšia podobnosť je na hlavnej diagonále, podľa ktorej je matica $\boldsymbol{S}$ súmerná. Táto reprezentácia odhaľuje vzťahy medzi časovými segmentami nahrávok a možno ju využiť pre rozbor sekvencií hudobných nahrávok alebo rozpoznávanie skladieb.



Obr. 1.6: Miera vlastnej podobnosti nahrávky ruchu nákupného centra.

2 TECHNIKY HĹBKOVÉHO UČENIA

2.1 Strojové učenie

2.1.1 Úvod

Teoretické základy princípov strojového učenia rozoberané v tejto časti sú potrebné pre pochopenie hĺbkových neurónových sietí využívaných v práci pre klasifikáciu. Základná požiadavka na algoritmus strojového učenia je naučiť sa podľa určitej skúsenosti plniť špecifickú úlohu, pričom výkonnosť algoritmu sa s pribúdajúcou skúsenosťou zlepšuje [5]. Takto naučené algoritmy majú využitie v mnohých odvetviach pre vykonávanie nasledujúcich základných úloh [6]:

- Klasifikácia
- Regresia
- Transkripcia
- Strojový preklad
- Štrukturalizácia výstupu
- Detekcia anomálií
- Syntéza a vzorkovanie
- Doplnenie chýbajúcich hodnôt
- Odstránenie šumu
- Odhad hustoty pravdepodobnosti

Pre posúdenie úspešnosti učenia nás zaujíma akú presnosť je algoritmus schopný dosiahnuť na vstupných dátach, ktoré zatiaľ nevidel v trénovacej fáze. Preto vstupné dáta zvyčajne rozdeľujeme na trénovaciu a testovaciu množinu [6]. Spracovaním výstupov modelu môžeme následne určiť rôzne ukazatele presnosti popísané v časti 2.5.4.

2.1.2 Učenie bez učiteľa

Tento druh strojového učenia často nazývaný tiež zhlukovanie alebo klastrovanie sa používa v aplikáciach, kedy algoritmus nemá presne určené triedy výstupov. Vstupné dáta týchto algoritmov často obsahujú veľké množstvo vlastností, pričom úlohou algoritmu je nájsť medzi nimi isté súvislosti. Na základe týchto súvislostí následne dáta štrukturalizuje a rozdeľuje do samostatných množín (zhlukov).

2.1.3 Učenie s učiteľom

V prípade, že očakávaným výstupom algoritmu strojového učenia je zaradenie do určitej dátovej triedy, hovoríme o učení s učiteľom alebo klasifikácii. V trénovacej fázy sú vstupné dáta označené a algoritmus má teda informáciu o tom, na akú triedu sa zrovna učí. V následnom použití sú algoritmu predložené neznáme dáta, o ktorých musí rozhodnúť, či patria do príslušnej triedy alebo nie.

2.1.4 Lineárna regresia

Príklad jednoduchého algoritmu strojového učenia je lineárna regresia používaná na predpoveď neznámych hodnôt v súbore bodov. Pokiaľ uvažujeme vektor vstupných hodnôt $\boldsymbol{x} \in \mathbb{R}$ a hodnotu výstupu $\widehat{y} \in \mathbb{R}$ môžeme zapísať lineárnu funkciu regresie následovne.

$$\widehat{y} = \boldsymbol{w}\boldsymbol{x} \tag{2.1}$$

Závislosť medzi vstupnou hodnotou x_i a výstupnou predpoveďou $\widehat{y}$ určuje vektor parametrov $\boldsymbol{w}$. V prípade, že vstupné hodnoty sú násobené kladným alebo záporným koeficientom w_i, hodnota výstupu narastá alebo klesá. Ak je koeficient nulový, nemá na výstup žiaden vplyv. Pre príklad uvažujeme súbor vstupných hodnôt s dĺžkou m, ku ktorým poznáme ich korektnú hodnotu výstupu y. Ako ukazovateľ presnosti predikcie použijeme strednú kvadratickú odchýlku MSE podľa následovnej závislosti.

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(\widehat{y}_i - y_i)^2 \tag{2.2}$$

Odchýlka sa teda rovná nule v prípade že predikované výstupné hodnoty sa rovnajú korektným výstupným hodnotám. Snahou je preto nájsť taký súbor parametrov $\boldsymbol{w}$, pri ktorom bude stredná kvadratická chyba minimálna [6].

2.1.5 Kapacita algoritmu

V prípade, že je možné efektívnejšie modelovať súbor bodov inak ako lineárnou závislosťou, dostávame polynóm n-tého rádu, kde parameter b určuje lineárny posun.

$$\widehat{y} = b + \sum_{i=0}^{n} w_i x^i \tag{2.3}$$

Obmedzenie algoritmu strojového učenia na určitú kapacitu zabraňuje nesprávnemu modelovaniu trénovacích dát, kedy môžu nastať dve situácie tzv. *underfitting* a *overfitting*. *Underfitting* nastáva v prípade, že algoritmus nemá dostatočnú kapacitu

dosiahnuť nízku trénovaciu odchýlku. V opačnom extrémnom prípade nastáva *overfitting*, kedy je výsledný model príliš presný a zachytáva náhodný šum v trénovacej množine. Takýto algoritmus vykazuje veľmi nízku testovaciu odchýlku, no výsledky na neznámych testovacích dátach sú nepresné [6].

2.2 Umelá neurónová sieť

2.2.1 Úvod

Motiváciou pre umelé neuronové siete sú biologické systémy tvorené jednoduchými samostatnými neurónmi prepojenými medzi sebou a navzájom spolupracujúcimi. Informácia je prenášaná neurónovými spojeniami, pričom najkratší známy čas prepojenia sa pohybuje v rádoch 10^{-3} sekundy [5]. V porovnaní s výpočtovou rýchlosťou dnešných počítačov to je pomerne nízka rýchlosť, no napriek tomu je ľudský mozog schopný spracovať udalosti veľmi rýchlo. Táto skutočnosť vedie k domnienkam, že proces spracovania informácie je v biologických neurónových sieťach distribuovaný do paralelných operácií cez veľké množstvo neurónov. Túto podstatu napodobňujú umelé neuronové siete.

2.2.2 Umelý neurón

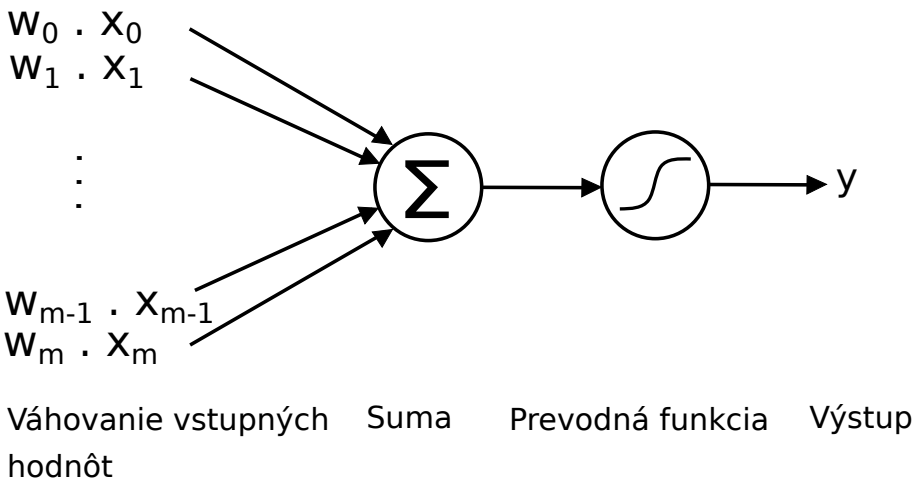
Vstupom umelých neurónov podobne ako v časti 2.1.4, je množina reálnych hodnôt $\boldsymbol{x}$, ktoré sú váhované vektorom koeficientov $\boldsymbol{w}$ a následne sčítané. Aktivácia výstupu závisí na prevodnej funkcii neurónu. Často používaná je sigmoidná funkcia s následovným predpisom.

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{2.4}$$

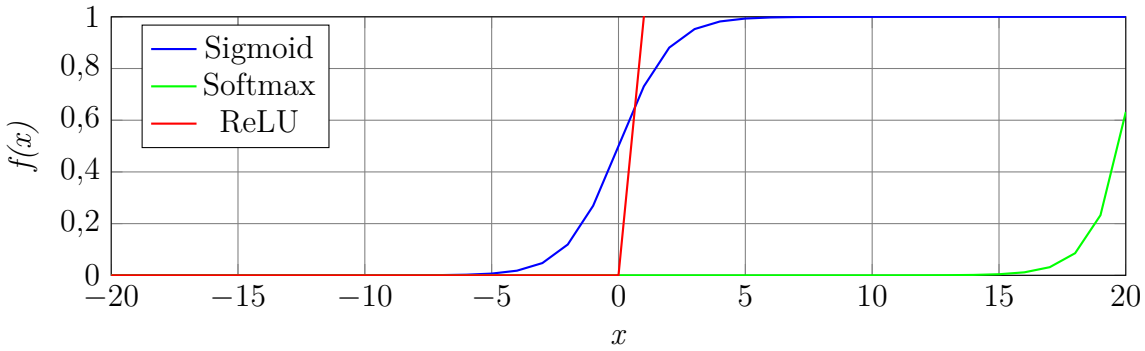
Výstup neurónu y potom nadobúda hodnoty v rozmedzí od 0 do 1 podľa následovného vzťahu.

$$y = \sigma(\sum_{i=0}^{m} w_i x_i) \tag{2.5}$$

Výstupy jednotlivých neurónov môžu byť spojené so vstupmi ďalších neurónov a tým vzniká umelá neurónová sieť. Pre nastavenie vstupných váh sa používajú tzv. *backpropagation* algoritmy, ktoré počas trénovacej fáze minimalizujú odchýlku porovnávaním predikovaných a reálnych výstupov [5].



Obr. 2.1: Výpočet aktivácie umelého neurónu [2].



Obr. 2.2: Porovnanie rôznych aktivačných funkcií.

2.2.3 Hlboká neurónová sieť

Neurónovú sieť v reálnom použití môže tvoriť vstupná vrstva, ktorá priamo sprostredkúva spojenie so vstupnými dátami, skrytá vrstva a výstupná vrstva, ktorá odovzdáva ďalej výsledky procesu. Neuróny v skrytej vrstve sú často hlavnými strojcami algoritmu a formujú sa do zložitých tvarov s mnohými prepojeniami [2]. Preto hovoríme o hlbokom učení.

2.2.4 Dropout

Pri trénovaní veľkých sietí s viacerými skrytými vrstvami a mnohými neurónovými prepojeniami je vyššia pravdepodobnosť, že nastane pretrénovanie siete, teda *overfitting*. Tento problém možno eliminovať pomocou techniky *dropout*. Hlavnou myšlienkou je náhodne odstrániť niektoré neuróny spolu s ich prepojeniami zo systému počas trénovania, čo zabraňuje jeho prílišnej adaptácii [7].

2.3 Konvolučný model

Pri rozpoznávaní objektov v obraze je potrebné dvojrozmerný signál previesť na vektor príznakov, ktorý predstavuje dáta pre skrytú vrstvu neurónovej siete. Tento prevod realizuje konvolučný model zložený z viacerých blokov a konvolučných vrstiev. Postupným vrstvením výstupov jednotlivých konvolučných filtrov a zmenšovaním rozlíšenia je vytvorený vektor príznakov. Konvolučný model obsahuje trénovateľné parametre, to znamená, že sa dokáže naučiť, ktoré artefakty obrazu sú dôležité pre rozpoznanie danej triedy.

2.3.1 Konvolučná vrstva

Každý blok modelu tvorí viacero priestorových filtrov s rôznymi koeficientami filtračného jadra s podstatne menším rozlíšením. Princíp priestorového filtrovania je založený na jednoduchom posuve jadra cez všetky body obrazu. V každom bode sú sumarizované výsledky násobenia hodnôt jasu pixelov koeficientami prekrývajúceho jadra. Koeficienty w filtračného jadra v tabuľke 3.12 vyjadríme ako funkčné hodnoty dvojrozmerného signálu.

Tab. 2.1: Koeficienty filtračného jadra[8].

$w(-1,-1)$	$w(-1,0)$	$w(-1,1)$
$w(0,-1)$	$w(0,0)$	$w(0,1)$
$w(1,-1)$	$w(1,0)$	$w(1,1)$

Následne možno matematicky popísať odozvu signálu f s rozlíšením M×N v jednom bode na filtračné jadro s rozmerom m×n podľa vzťahu 3.3.

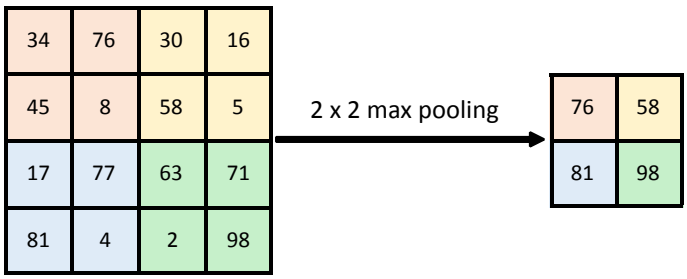
$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} w(s,t) f(x+s, y+t) \tag{2.6}$$

Z princípu vyplýva, že parametre budú $a = (m-1)/2$ a $b = (n-1)/2$. Aby sme získali kompletný dvojrozmerný obraz je potrebné prejsť všetky body vstupného obrazu pre $x = 0, 1, 2, ..., M-1$ a $y = 0, 1, 2, ..., N-1$. Zmenou koeficientov filtračného

jadra je možné detekovať konkrétne vlastnosti obrazu ako horizontálne a vertikálne hrany alebo elementárne tvary [8].

2.3.2 Podvzorkovanie

Zmenšenie rozlíšenia obrazu je realizované zredukovaním počtu pixelov. Podľa princípu výpočtu výstupnej hodnoty v rámci okna rozdeľujeme algoritmy *max pooling* 2.3 a *average pooling*. Pre dosiahnutie polovičného rozlíšenia posúvame okno veľkosti 2×2 cez celý obraz bez prekrytia. Metóda *max pooling* jednoducho vyberá lokálne maximum signálu, *average pooling* počíta priemer všetkých pixelov v rámci okna.



Obr. 2.3: Príklad výberu maxima algoritmom *max pooling*.

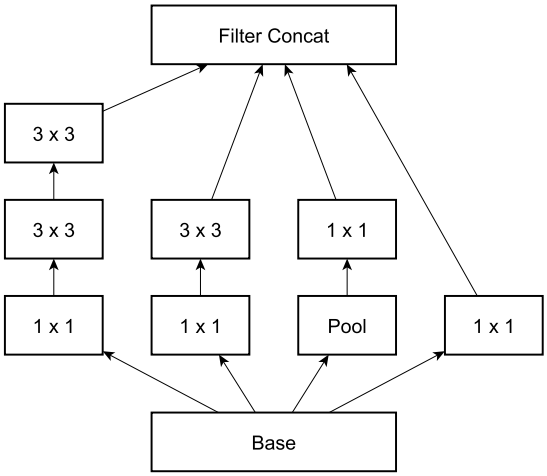
2.3.3 Model VGG16

Konvolučný model VGG16 je zostavený z piatich blokov. Každý blok obsahuje konvolučné vrstvy a vrstvu *max pooling*. Veľkosť filtračného jadra vo všetkých konvolučných vrstvách je podľa [9] 3×3. Celková architektúra modelu je zobrazená na obrázku 2.4. Všetky skryté konvolučné vrstvy sú vybavené funkciou *ReLu*, čím je spôsobená nelinearita na výstupe. Rozlíšenie vstupného obrazu musí byť 244×244. Celkový počet vykonaných operácií pri prechode obrazu modelom je podľa [10] približne $32 \cdot 10^9$.

Obr. 2.4: Architektúra konvolučného modelu VGG16 [9].

2.3.4 Model InceptionV3

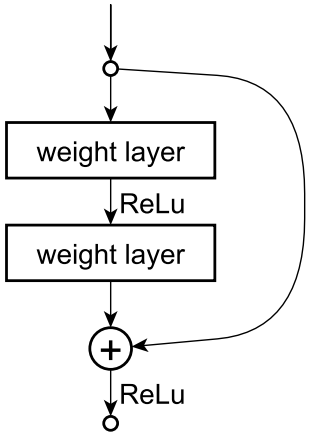
Tretia verzia modelu InceptionV3 vychádza z architektúry predošlých verzií Inception. Zníženie počtu trénovacích parametrov druhej verzie oproti prvej je dosiahnuté nahradením filtru s jadrom 5×5 dvoma za sebou radenými filtrami s jadrom 3×3. V tretej verzií je naviac doplnená normalizácia trénovacej dávky *batch normalization* v pomocnom klasifikátore podľa [11]. Model predpokladá vstupné rozlíšenie obrazu 299×299. Počet operácií pri spracovaní je podľa [10] podstatne nižší ako v prípade VGG16 a to približne $8 \cdot 10^9$.



Obr. 2.5: Architektúra jedného z modulov modelu InceptionV3 [11].

2.3.5 Model ResNet18

Tretím modelom použitým v práci je model ResNet18, ktorý obsahuje najmenej trénovacích parametrov zo všetkých testovaných modelov a preto spracuje obraz v najnižšom čase. Na rozdiel od ostatných modelov, ResNet využíva architektúru sieť v sieti. To znamená, že vstup je spojený s výstupom dvoch vnútorných vrstiev a ďalej generovaný cez aktivačnú funkciu *ReLu* [12]. Príklad tejto štruktúry je zobrazený na obrázku 2.6. ResNet18 je zložený z osemnástich konvolučných vrstiev a podľa [10] vykoná približne $3 \cdot 10^9$ operácií pri spracovaní jedného snímku.



Obr. 2.6: Príklad štruktúry modelu ResNet [12].

2.4 Technológie pre prácu s hlbokými sieťami

2.4.1 TensorFlow

Framework TensorFlow bol vyvíjaný spoločnosťou Google, ktorá ho sprístupnila pod voľnou licenciou v roku 2015. Podľa autorov [13] je TensorFlow systém pre experimentovanie s novými modelmi, trénovanie na veľkých dátových množinách a zavedenie do produkcie. Pomocou nástroja TensorBoard je možné zobraziť grafy stavu trénovania v reálnom čase a schému dátových tokov medzi jednotlivými blokmi, čo pomáha optimalizovať trénovacie algoritmy. V TensorFlow reprezentuje každú dátovú jednotku n-dimenzionálne pole, čím je modelovaný tenzor. Tenzory sú vstupmi a výstupmi matematických operácií algoritmu a môžu nadobúdať premenlivú dĺžku.

2.4.2 Keras

Keras je oficiálna API nadstavba pre zjednodušenie práce s výpočtovými nástrojmi hlbokého učenia vrátane TensorFlow. Knižnica Keras v sebe implementuje tiež nástroje pre prácu so vstupnými dátami a umožňuje automatizovať augmentáciu. Hlavným stavebným blokom je *model* a najjednoduchším modelom je *sequential* predstavujúci lineárne zoskupenie neurónových vrstiev [14]. Ujasnenie významu nasledujúcich parametrov je dôležité pre správne nastavenie trénovacieho algoritmu.

- *epochs* - koľkokrát bude opakovane celý trénovací dataset spracovaný sieťou.
- *batch size* - počet spracovaných trénovacích jednotiek, po ktorých sa aktualizujú váhy prepojení.
- *optimizer* - špecifický algoritmus, ktorý aktualizuje váhy prepojení.

2.5 Ohodnotenie modelu

2.5.1 Binárna klasifikácia

V mnohých prípadoch je úlohou algoritmu strojového učenia rozdeliť množinu vstupných dát do dvoch výstupných tried, preto v tomto prípade hovoríme o binárnej klasifikácií. Pre vyhodnotenie úspešnosti tejto techniky môžeme použiť ROC analýzu používanú počas druhej svetovej vojny na odlíšenie radarových signálov skutočných odrazov a šumu. Dnes sa ROC grafy využívajú najmä v medicíne a strojovom učení na vizualizáciu a analýzu správania rozhodovacích algoritmov [15]. Pre ďalší popis budeme preto predpokladať dva stavy označenia výstupu ako pozitívny alebo negatívny.

2.5.2 Matica zámen

Podľa zaradenia vstupného objektu algoritmom strojového učenia do triedy môžeme rozdeliť jednotlivé prípady do následovných skupín.

- *True positive* (TP) - pozitívny prvok je vyhodnotený ako pozitívny.
- *False negative* (FN) - pozitívny prvok je vyhodnotený ako negatívny.
- *True negative* (TN) - negatívny prvok je vyhodnotený ako negatívny.
- *False positive* (FP) - negatívny prvok je vyhodnotený ako pozitívny.

Pre binárnu klasifikáciu má matica zámen 2.2 rozmer 2×2, pričom riadky reprezentujú predikované a stĺpce skutočné triedy. Pri správne nastavenej rozhodovacej úrovni sa najvyššie hodnoty nachádzajú na hlavnej diagonále a možno z nich vypočítať niektoré ukazovatele výkonnosti rozhodovacieho algoritmu.

Tab. 2.2: Príklad zobrazenia matice zámen.

		Skutočnosť	
		Trieda A	Trieda B
Predikcia	Trieda A	TP	FP
	Trieda B	FN	TN

2.5.3 Citlivosť a rozlíšiteľnosť

Krivka ROC predstavuje závislosť medzi citlivosťou a rozlíšiteľnosťou rozhodovacieho algoritmu podľa nastavenia rozhodovacieho prahu. V prípade nastavenia prahu príliš nízko dosiahneme vysokú citlivosť, no rozlíšiteľnosť rozhodovania bude malá a jednotlivé triedy sa budú medzi sebou miešať. Naopak, v prípade že je rozhodovacia úroveň príliš vysoko, nízka citlivosť spôsobí, že algoritmus prehliadne niektoré pozitívne prvky. Citlivosť v anglickej literatúre [15] nazývaná *true positive rate* (TPR) predstavuje pomer počtu správne vyhodnotených pozitívnych prvkov voči počtu všetkých pozitívnych prvkov a vynáša sa na zvislú os ROC diagramu. Na vodorovnú os sa vynáša rozlíšiteľnosť ako jednotkový doplnok tzv. *false positive rate* (FPR).

$$TPR = \frac{TP}{TP + FN} \tag{2.7}$$

$$FPR = \frac{TN}{TN + FP} \tag{2.8}$$

Často je uprednostnená vyššia rozlíšiteľnosť pred citlivosťou z dôvodu, aby dochádzalo k čo najmenšiemu počtu falošných detekcií pozitívnych prvkov.

2.5.4 Úspešnosť rozpoznania

Podľa literatúry [15] je základným skalárnym vyjadrením úspešnosti rozpoznania plocha pod krivkou ROC. V prípade, že sa AUC[1] blíži k hodnote 1 je úspešnosť vysoká. Ďalšie používané ukazovatele výkonnosti ako presnosť rozpoznania pozitívneho prvku PPV[2] a presnosť rozpoznania negatívneho prvku NPV[3] sú závislé na zmene rozloženia pravdepodobnosti zaradenia do jednotlivých tried. V anglickej literatúre je presnosť PPV často nazývaná ako *precision*. Hodnoty PPV a NPV je možné vypočítať podľa následovných vzťahov.

$$PPV = \frac{TP}{TP + FP} \tag{2.9}$$

$$NPV = \frac{TN}{TN + FN} \tag{2.10}$$

Celková presnosť ACC[4] v literatúre nazývaná často *accuracy* určuje pomer správne zaradených prvkov k počtu všetkých prvkov testovacej množiny.

$$ACC = \frac{TP + TN}{TN + TP + FN + FP} \tag{2.11}$$

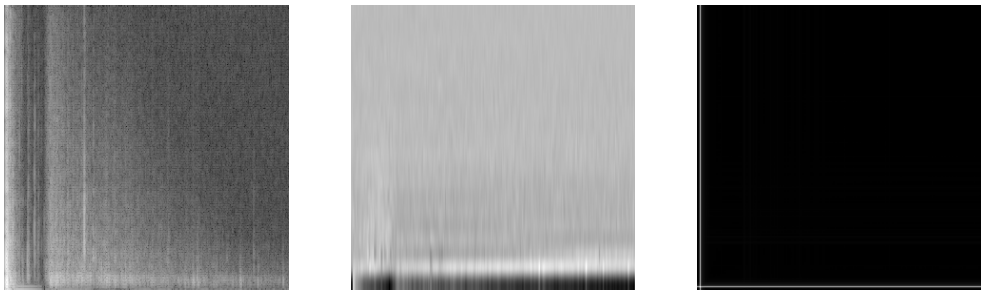
[1] AUC - Area Under Curve
[2] PPV - Positive Predictive Value
[3] NPV - Negative Predictive Value
[4] ACC - Accuracy

3 PRAKTICKÁ IMPLEMENTÁCIA

3.1 Extrakcia príznakov

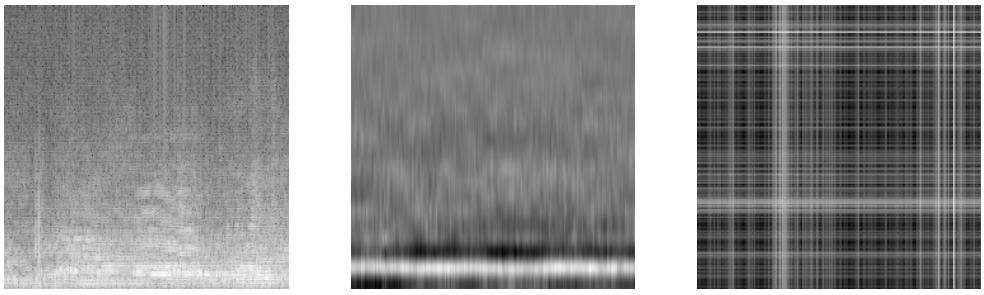
3.1.1 Základný koncept

Podľa štúdie [8] je možné spoľahlivo aplikovať technológiu rozpoznávania a klasifikácie obrazu taktiež na zvukové dáta. Dosiahnutá presnosť však závisí najmä od spôsobu vizualizácie zvuku. Je možné kombinovať viaceré obrazové reprezentácie popísané v časti 1.2.1 a porovnať dosiahnuté výsledky. V tejto práci sú obrazy použité pre trénovanie neurónovej siete zložené z troch vrstiev RGB[1]. Každý farebný kanál obsahuje inú vizualizáciu zvuku a výsledný obrazec tak nesie väčšie množstvo informácií.Poradie kanálov je zobrazené na obrázku 3.1. Zobrazený je spektrogram, MFCC koeficienty a matica podobností zvuku výstrelu zo zbrane.



(a) Spektrogram (b) MFCC koeficienty (c) Matica podobnosti

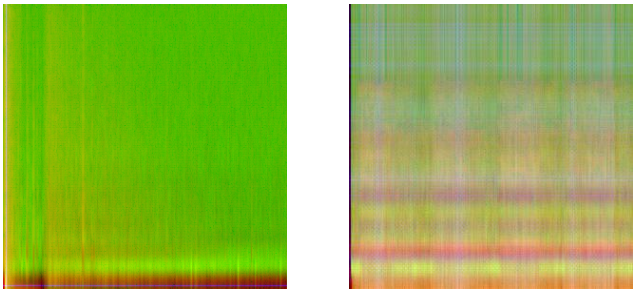
Obr. 3.1: Farebné kanály RGB zvuku výstrelu.



(a) Spektrogram (b) MFCC koeficienty (c) Matica podobnosti

Obr. 3.2: Farebné kanály RGB zvuku pozadia.

[1]RGB - Red Green Blue



(a) Výstrel zo zbrane (b) Ruch ulice

Obr. 3.3: Výsledná vizualizácia zvukov ako RGB obraz.

Podľa štúdie [8] dosahuje v klasifikácií najlepšie výsledky použitie spektrogramu a MFCC. V treťom farebnom kanály je použitá matica vlastnej podobnosti. V prípade rozpoznávania zvuku výstrelu od náhodného pozadia existuje predpoklad, že kombinácia týchto vizualizácií odhalí zvuky s odlišným charakterom od výstrelov. Po vykreslení matice podobnosti výstrelu vidíme len dve úzke čiary 3.1, no v prípade zvuku s pravidelnou periodicitou sa v obraze začne objavovať pravidelná mriežka 3.2. MFCC koeficienty môžu odhaliť zvuky rečového charakteru. Porovnanie výsledných RGB obrazov výstrelu a náhodného pozadia je zobrazené na obrázku 3.3.

3.1.2 Databáza nahrávok

Zvukové nahrávky výstrelov a náhodných pozadí boli získané z volne dostupných zvukových databáz. Databáza výstrelov má názov The Free Firearm Sound Library – Expanded Edition a obsahuje viac ako 1000 nahrávok v nekomprimovanom formáte WAV so vzorkovacou frekvenciou $f_s = 44,1\,kHz$. Jedná sa o krátke, priemerne 1 sekundu trvajúce nahrávky, ktoré sú podľa autorov banky Airborne Sound zbavené praskania a nežiadúceho šumového pozadia. Databáza nahrávok pozadí UrbanSound8K obsahuje viac ako 8000 zvukov z rôznych zdrojov najmä hluk ulice, detský krik, štekanie psov, sirény a iné hlukové pozadie. Pôvodne databáza obsahovala taktiež výstrely zo zbraní, tie však boli odstránené. Formát nahrávok je rovnaký ako v prípade výstrelov.

3.1.3 Spracovanie databáze

Obrazová databáza bola vygenerovaná v programe MATLAB. Pri spracovaní boli zdrojové nahrávky skrátené na úseky s dĺžkou trvania jednej sekundy a odfiltrované tie, ktoré požadovanú dĺžku nedosahovali. V rámci základného testu boli v následnom spracovaní vypočítané matice spektrogramu, MFCC koeficientov a vlastnej podobnosti. Dĺžka časového segmentu pri výpočte MFCC a spektrogramu je 256

vzorkov s polovičným prekrytím a váhovaním Hammingovím oknom. Počet koeficientov MFCC je 20. Zvislá os je v oboch vizualizáciách lineárna. Signál bol spracovaný v plnom vzorkovacom rozlíšení nahrávky a vložený do matice po časových úsekoch. Následne boli rozmery matice zmenšené na požadovaný rozmer, podľa použitého konvolučného modelu 2.3. Všetky vizualizácie boli normalizované tak, aby nadobúdali hodnoty v rozmedzí od 0 po 1.

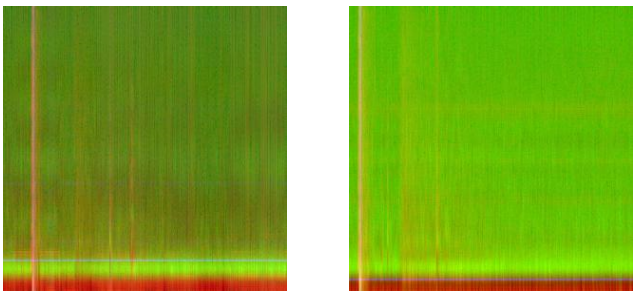
3.1.4 Časový posun a náhodné pozadie

V testoch boli generované vizualizácie výstrelov s pridaným pozadím a náhodným časovým posunom v rámci intervalu $\frac{1}{4}$. Takto je možné zväčšiť trénovaciu množinu dát a priblížiť testovanie k reálnejším podmienkam. Ak uvažujeme pozadie ako náhodný šum, odstup užitočného signálu od šumu SNR[2] môžeme vyčísliť pomerom efektívnych hodnôt RMS[3] podľa rovnice 3.1, kde x_n predstavuje postupnosť vzorkov signálu dĺžky N.

$$RMS = \sqrt{\frac{1}{N}\sum_{n=1}^{N} x_n^2} \tag{3.1}$$

$$SNR_{dB} = 10\log_{10}\left(\frac{RMS_{signál}}{RMS_{šum}}\right) \tag{3.2}$$

Porovnanie zobrazení výstrelu s rôznou hodnotou SNR je na obrázku 3.4. Minimálna hodnota SNR v testoch bola 0 dB kedy ešte bolo možné výstrel subjektívne rozpoznať ľudským uchom a nebol tak stratený v pozadí.



(a) SNR 0 dB (b) SNR 10 dB

Obr. 3.4: Porovnanie výstrelov s rôznym odstupom signálu od šumu.

[2]SNR - Signal to Noise Ratio
[3]RMS - Root Mean Square

3.2 Trénovanie neurónovej siete

3.2.1 Zostavenie siete

V tejto časti je potrebné zostaviť celkovú architektúru neurónovej siete. Jedná sa o prepojenie konvolučného modelu s plne prepojenou skrytou vrstvou a výstupnou vrstvou. Načítanie a zamknutie trénovania parametrov konvolučného modelu sa prevedie následovnými príkazmi.

```
vgg = VGG16(include_top=False,
            input_shape=(img_size, img_size, 3))

for layer in vgg.layers[:]:
        layer.trainable = False
```

Výstup konvolučného modelu je plne prepojený s vrstvou veľkosti 1024 s aktivačnou funkciou *ReLu*. Medzi výstupnou a skrytou vrstvou sa prevedie náhodné odstránenie neurónových prepojení *dropout*, popísaný v časti 2.2.4.

```
model = models.Sequential()
model.add(vgg)
model.add(layers.Flatten())
model.add(layers.Dense(1024, activation='relu'))
model.add(layers.Dropout(0.5))
model.add(layers.Dense(2, activation='softmax'))
```

3.2.2 Nastavenie parametrov

V následnej časti bolo potrebné nastaviť správne parametre učenia a prispôsobiť vstupné dáta. Vstupné obrázky sú označené triedou podľa umiestnenia v zložke trénovacej množiny. Hodnoty jasu sú škálované do rozmedzia od 0 do 1. Počet obrázkov *batchsize*, po ktorých budú nastavené váhy prepojení je nastavený na 10. V následovnej časti programu sú inicializované dátové generátory, ktoré sa starajú o načítanie obrázkov z priečinkov rozdelených na trénovaciu a validačnú množinu.

```
train_datagen = ImageDataGenerator(rescale=1./255)
train_gen = train_datagen.flow_from_directory(
        DIR + '/train',
        target_size=(img_size, img_size),
        batch_size=10,
        class_mode='categorical')
```

```
valid_datagen = ImageDataGenerator(rescale=1./255)
valid_gen = valid_datagen.flow_from_directory(
        DIR + '/valid',
        target_size=(img_size, img_size),
        batch_size=10,
        class_mode='categorical',
        shuffle=False)
```

Príprava modelu na trénovanie sa prevedie príkazom *compile*. Popri nastavení optimalizačného algoritmu je definovaný parameter *learning rate*, ktorý určuje mieru zmeny váh pri ich nastavovaní.

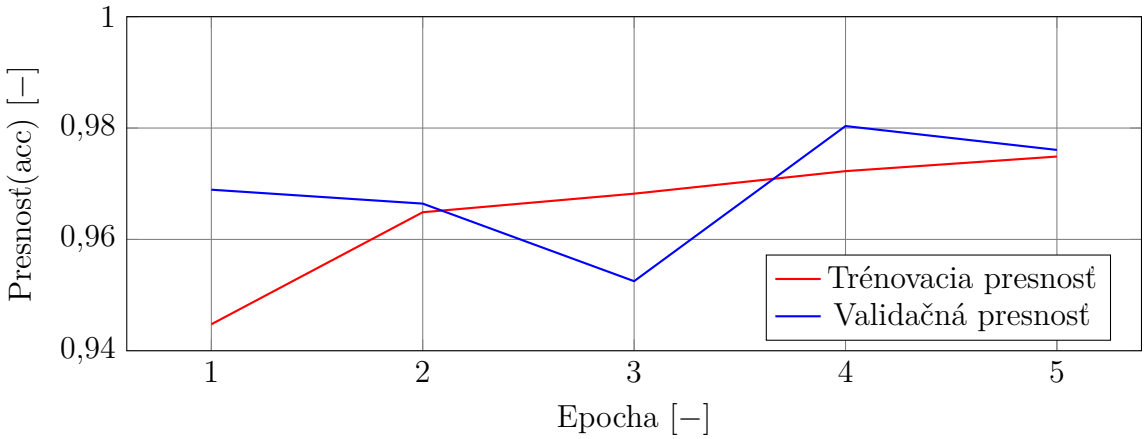
```
model.compile(loss='categorical_crossentropy',
              optimizer=optimizers.RMSprop(lr=1e-4),
              metrics=['acc'])
```

3.2.3 Priebeh trénovania

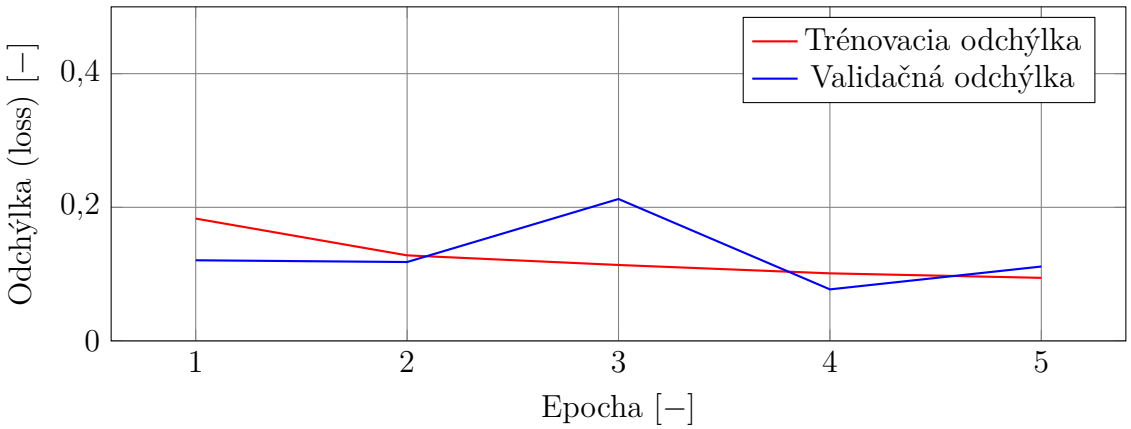
Kapacita algoritmu je obmedzená nastavením počtu trénovacích epoch na 5. Začiatok trénovania je spustený príkazom *fit_generator* s následovnými parametrami.

```
train_steps = train_gen.samples/train_gen.batch_size
valid_steps = valid_gen.samples/valid_gen.batch_size
model.fit_generator(train_gen,
                    steps_per_epoch=train_steps,
                    epochs=5,
                    validation_data=valid_gen,
                    validation_steps=valid_steps,
                    verbose=1)
```

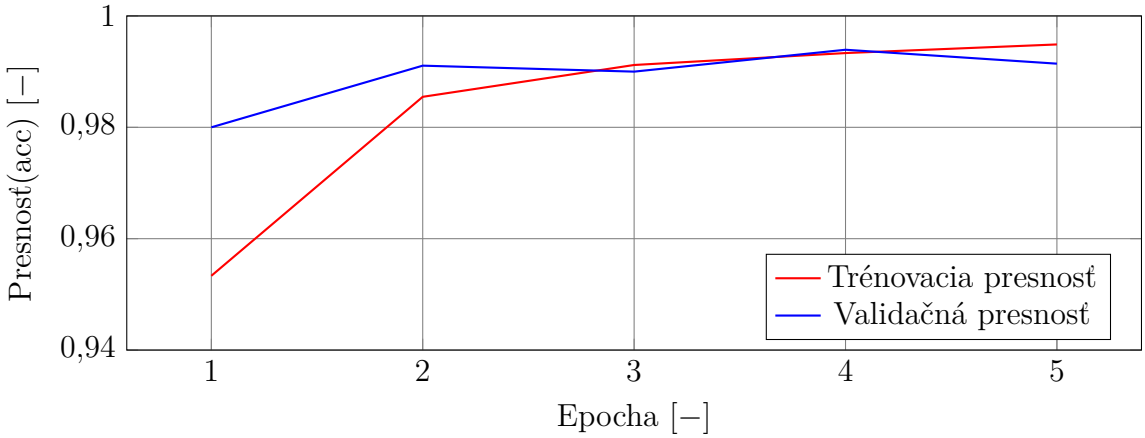
Následne začína postupné trénovanie pričom priebežne je možné sledovať aktuálnu trénovaciu a validačnú presnosť. V procese učenia sú okrem trénovacích dát sieti predkladané dáta z validačnej množiny, ktoré nepozná a tým sa proces približuje k reálnejším podmienkam. Zároveň je menej pravdepodobné, že nastane *overfitting*, podrobne vysvetlený v časti 2.1.5. Na priebehu trénovacej presnosti modelu v grafe 3.7 je vidno, že po štvrtej epoche sa presnosť prestáva zvyšovať. Použitím 4-jadrového CPU Intel Core i7 s ôsmimi vláknami a s taktovacou frekvenciou 2 GHz zabralo učenie siete niekoľko hodín. Celkovú dobu výpočtov je možné výrazne znížiť použitím grafických jadier.



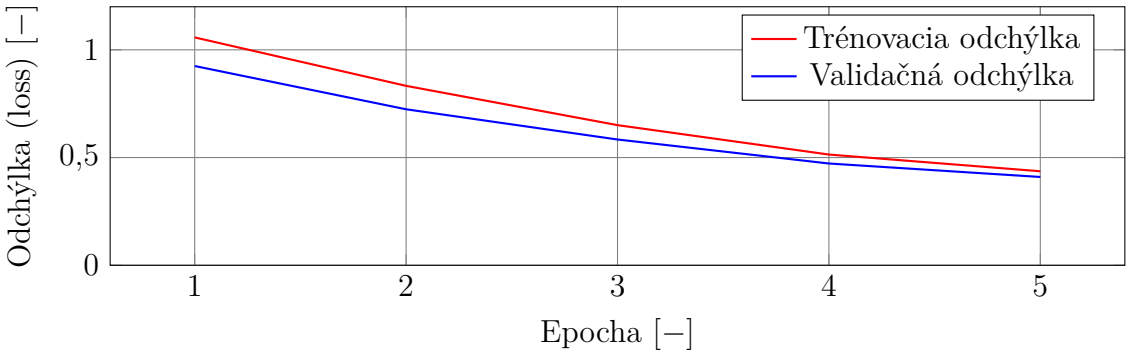
Obr. 3.5: Trénovacia a validačná presnosť počas trénovania modelu VGG16.



Obr. 3.6: Trénovacia a validačná odchýlka počas trénovania modelu VGG16.



Obr. 3.7: Trénovacia a validačná presnosť počas trénovania modelu ResNet18.



Obr. 3.8: Trénovacia a validačná odchýlka počas trénovania modelu ResNet18.

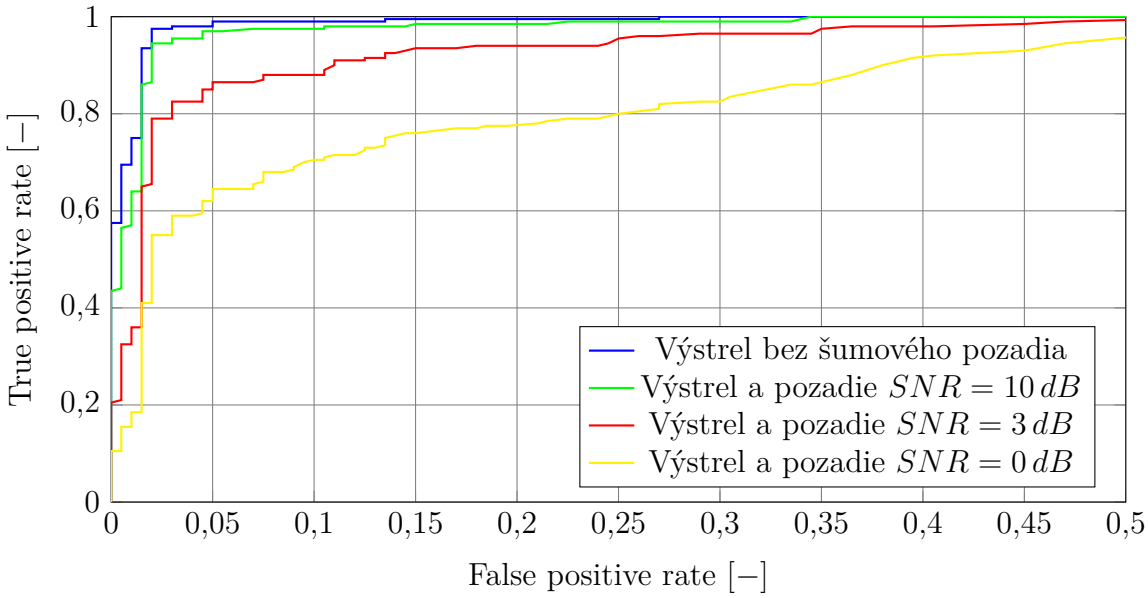
3.3 Testovacia fáza

Počas testovacej fázy boli na vstup natrénovaného modelu vložené obrázky, ktoré sa nevyskytovali v trénovacej množine. Výstupom je pravdepodobnosť zaradenia do jednotlivých tried medzi výstrely alebo pozadia. Podľa výsledného ROC grafu bola hľadaná taká rozhodovacia úroveň, pri ktorej čo najmenej výsledkov spadalo medzi falošné výstrely a zároveň bola dosiahnutá pomerne vysoká úspešnosť detekcie reálnych výstrelov. Následné výsledky porovnávajú rôzne postupy trénovania a spôsoby vizualizácie.

3.3.1 Trénovacia množina výstrelov s jednou hodnotou SNR

V tejto časti bola neuronová sieť natrénovaná na výstrely s veľmi prísnym odstupom od náhodného pozadia $SNR = 0\,dB$. Konvolučný model použitý v tomto teste je InceptionV3. Trénovacia množina obsahovala 800 a testovacia množina celkovo 200 obrázkov. Natrénovanej neurónovej sieti boli predkladané obrázky výstrelov s rôznou hodnotou SNR. Na obrázku 3.9 sú zobrazené ROC krivky jednotlivých testov. Úspešnosť rozpoznania, ktorú môžeme ohodnotiť obsahom pod krivkou, stúpa s hodnotou SNR. Podľa predpokladu dosahuje najvyššiu úspešnosť test bez pripočítania šumového pozadia a naopak, najhoršie výsledky dosiahlo rozpoznávanie výstrelov so započítaním šumom s odstupom $SNR = 0\,dB$.

Následné tabuľky zobrazujú matice zámen jednotlivých testov. Rozhodovacie úrovne boli zvolené podľa kriviek ROC tak, aby čo najmenej pozadí bolo vyhodnotených ako falošný výstrel. V prípade testov so šumovým pozadím je na úkor tejto požiadavky vysoký počet výstrelov vyhodnotených ako pozadie. Prípustná chybovosť falošných výstrelov FPR bola stanovená 1 %. Keďže testovacia množina pozadí sa nemenila, rozhodovacia úroveň 0,85 odpovedajúca danej chybovosti ostáva rovnaká pre všetky testy.



Obr. 3.9: ROC krivky testovania modelu trénovaného na jednu hodnotu $SNR = 0\,dB$.

Tab. 3.1: Matica zámen modelu trénovaného výstrelmi s hodnotu $SNR = 0\,dB$ a testovaného výstrelmi bez šumového pozadia.

		Skutočnosť	
		Výstrel	Pozadie
Predikcia	Výstrel	148	2
	Pozadie	52	198

Tab. 3.2: Matica zámen modelu trénovaného výstrelmi s hodnotu $SNR = 0\,dB$ a testovaného výstrelmi s hodnotu $SNR = 10\,dB$.

		Skutočnosť	
		Výstrel	Pozadie
Predikcia	Výstrel	127	2
	Pozadie	73	198

Tab. 3.3: Matica zámen modelu trénovaného výstrelmi s hodnotu $SNR = 0\,dB$ a testovaného výstrelmi s hodnotu $SNR = 3\,dB$.

		Skutočnosť	
		Výstrel	Pozadie
Predikcia	Výstrel	70	2
	Pozadie	130	198

Tab. 3.4: Matica zámen modelu trénovaného výstrelmi s hodnotu $SNR = 0\,dB$ a testovaného výstrelmi s hodnotu $SNR = 0\,dB$.

		Skutočnosť	
		Výstrel	Pozadie
Predikcia	Výstrel	37	2
	Pozadie	163	198

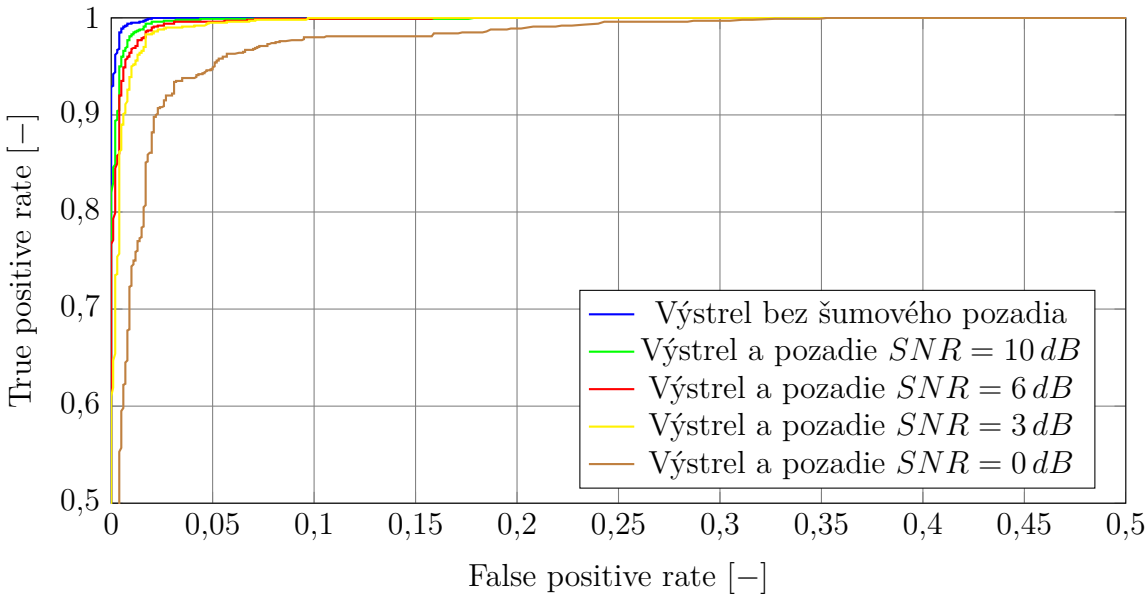
Porovnaním výsledkov v tabuľke 3.5 je možné sledovať, ako sa postupným zvyšovaním SNR testovacej množiny znižuje počet nerozpoznaných výstrelov a tým narastá celková presnosť rozpoznania ACC. Keďže stále menej výstrelov je vyhodnotených ako falošné pozadie, stúpa taktiež presnosť rozpoznania pozadia NPV. Takto natrénovaný model však nieje príliš presný ani pre výstrely bez pridaného šumu, kedy celková presnosť dosiahla 86,5 %.

Tab. 3.5: Porovnanie úspešnosti rozpoznania pri trénovaní modelu výstrelmi s jednou hodnotou $SNR = 0\,dB$.

Testovacie dáta	AUC	ACC	PPV	NPV	TPR	FPR
Výstrel bez šumového pozadia	0,993	0,865	0,987	0,792	0,740	0,01
Výstrel a pozadie SNR 10 dB	0,987	0,813	0,984	0,731	0,635	0,01
Výstrel a pozadie SNR 3 dB	0,959	0,670	0,972	0,604	0,350	0,01
Výstrel a pozadie SNR 0 dB	0,891	0,588	0,949	0,548	0,185	0,01

3.3.2 Trénovacia množina výstrelov so zmiešanými hodnotami SNR

V následnom teste bola neurónová sieť natrénovaná na výstrely s náhodnou hodnotou odstupu od šumového pozadia. Trénovacia množina teda obsahuje 6000 nahrávok s hodnotami $SNR = 0\,dB, 3\,dB, 6\,dB, 10\,dB$ a nahrávky bez pridaného pozadia. Jednotlivé testovacie množiny obsahujú vždy 1000 nahrávok s konkrétnou hodnotou SNR. Použitý konvolučný model je opäť InceptionV3. Keďže výsledky tohto testu sú výrazne lepšie ako v predošlom prípade, je možné určiť nižšiu toleranciu falošných výstrelov 0,5 % z celkového počtu testovacích pozadí. Podobne ako v predošlom teste je možné sledovať klesajúcu presnosť rozpoznania výstrelu pri nízkych odstupoch od pozadia.



Obr. 3.10: ROC krivky testovania modelu trénovaného na rôzne hodnoty SNR.

Následné tabuľky zobrazujú matice zámen jednotlivých testov. Optimálna rozhodovacia úroveň pre požadovanú chybovosť je 0,89. Táto úroveň ostáva rovnaká pre všetky testy, keďže množina testovacích pozadí sa nemení. Takto natrénovaná neuronová sieť vykazuje prekvapivo dobré výsledky taktiež pre výstrely s minimálnym odstupom od šumového pozadia. Na výsledkoch v tabuľke 3.11 vidno, že tento postup je odolnejší voči pridanému šumu, keďže výrazný skok nastáva až pre hodnoty SNR menšie ako 3 dB, kedy presnosť klesne pod hranicu 80%.

Tab. 3.6: Matica zámen modelu trénovaného výstrelmi s rôznou hodnotou SNR a testovaného výstrelmi bez šumového pozadia.

		Skutočnosť	
		Výstrel	Pozadie
Predikcia	Výstrel	989	5
	Pozadie	11	995

Tab. 3.7: Matica zámen modelu trénovaného výstrelmi s rôznou hodnotou SNR a testovaného výstrelmi s hodnotu $SNR = 10\,dB$.

		Skutočnosť	
		Výstrel	Pozadie
Predikcia	Výstrel	956	5
	Pozadie	44	995

Tab. 3.8: Matica zámen modelu trénovaného výstrelmi s rôznou hodnotou SNR a testovaného výstrelmi s hodnotu $SNR = 6\,dB$.

		Skutočnosť	
		Výstrel	Pozadie
Predikcia	Výstrel	930	5
	Pozadie	70	995

Tab. 3.9: Matica zámen modelu trénovaného výstrelmi s rôznou hodnotou SNR a testovaného výstrelmi s hodnotu $SNR = 3\,dB$.

		Skutočnosť	
		Výstrel	Pozadie
Predikcia	Výstrel	883	5
	Pozadie	117	995

Tab. 3.10: Matica zámen modelu trénovaného výstrelmi s rôznou hodnotou SNR a testovaného výstrelmi s hodnotu $SNR = 0\,dB$.

		Skutočnosť	
		Výstrel	Pozadie
Predikcia	Výstrel	583	5
	Pozadie	417	995

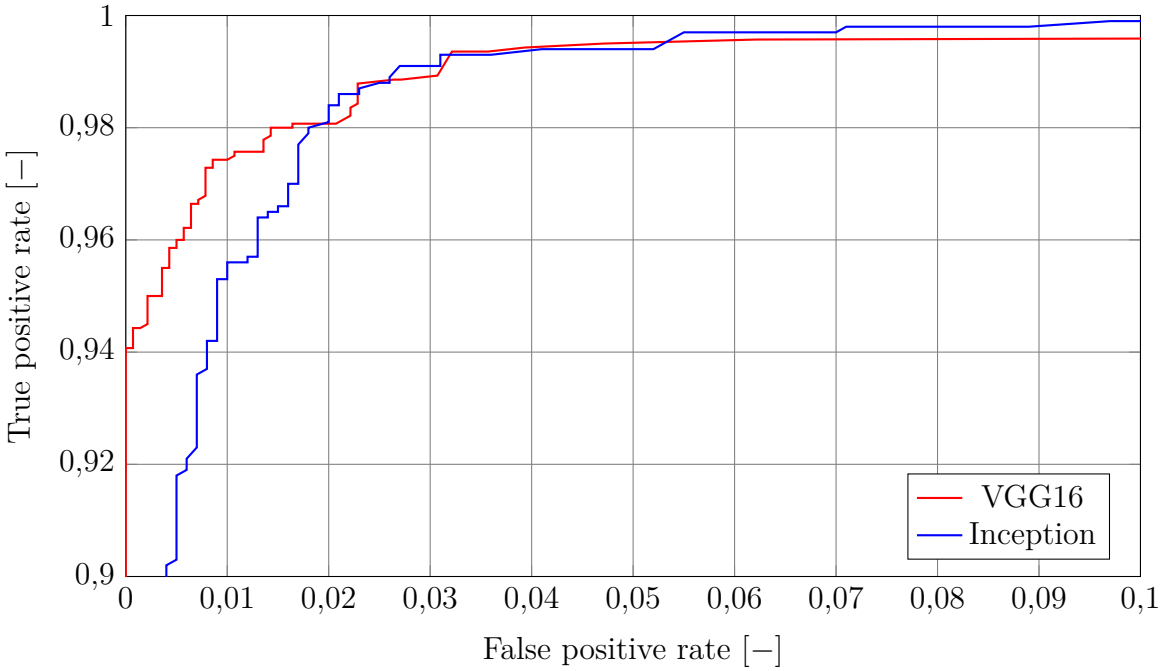
Tab. 3.11: Porovnanie úspešnosti rozpoznania pri trénovaní modelu výstrelmi s rôznou hodnotou SNR.

Testovacie dáta	AUC	ACC	PPV	NPV	TPR	FPR
Výstrel bez šumového pozadia	0,9997	0,992	0,995	0,989	0,989	0,005
Výstrel a pozadie SNR 10 dB	0,9990	0,976	0,995	0,958	0,956	0,005
Výstrel a pozadie SNR 6 dB	0,9984	0,963	0,995	0,934	0,930	0,005
Výstrel a pozadie SNR 3 dB	0,9974	0,939	0,994	0,895	0,883	0,005
Výstrel a pozadie SNR 0 dB	0,9872	0,789	0,991	0,705	0,583	0,005

3.4 Porovnanie rôznych konvolučných modelov

V tejto časti práce boli porovnané úspešnosti výberu dôležitých vlastností vizualizácie výstrelu pomocou dvoch konvolučných modelov VGG16 a InceptionV3. V oboch prípadoch je konvolučný model uzamknutý a trénované sú len plne prepojená a výstupná vrstva. Použité sú váhy neurónových prepojení predtrénované na databáze obrázkov ImageNet, ktorá neobsahuje žiadne vizualizácie zvuku. Práve kvôli tomu je ťažko predpokladať, ktorý z modelov si lepšie poradí s rozpoznaním vizualizácie výstrelu.

Databáza nahrávok bola opäť rozdelená na trénovaciu, testovaciu a validačnú množinu v pomere 60 %, 20 % a 20 %. Celkové množstvo vygenerovaných obrázkov bolo 7000 výstrelov a 7000 náhodných pozadí. Z predošlých výsledkov vyplýva, že neuronová sieť dosahuje vyššiu presnosť v prípade, že je natrénovaná na rôzne odstupy signálu od šumu. Preto boli k zvukom výstrelov opäť pripočítané zvuky pozadí s náhodným hodnotami SNR.



Obr. 3.11: Porovnanie modelov VGG16 a Inception podľa ROC kriviek.

Krivky ROC 3.11 naznačujú, že modely dosahujú rovnakú presnosť pri špecificite vyššej ako 0,02. V prípade, že požadujeme nižší pomer falošne pozitívnych predikcií ako 2 %, model VGG16 sa stáva vhodnejším, keďže príslušná krivka klesá menej strmo ako v prípade modelu Inception. Hodnoty v tabuľke 3.12 a matica zámen 3.13 sú počítané pre rovnakú chybovosť tak, aby falošne predikovaných výstrelov bolo maximálne 0,5 % z celkovej testovacej množiny.

Tab. 3.12: Vyčíslenie úspešnosti rozpoznania pomocou VGG16 a Inception.

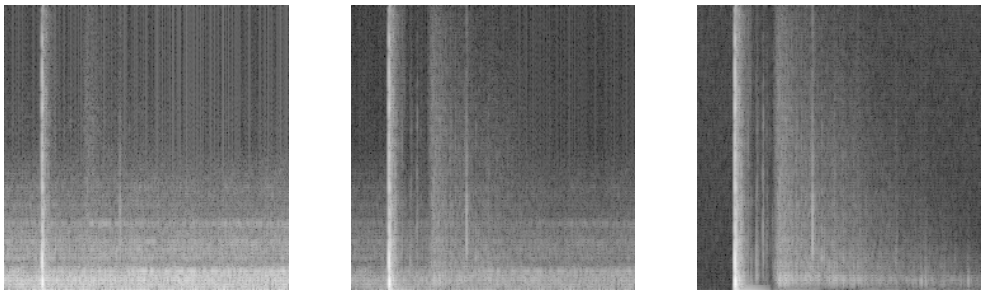
Testovaný model	AUC	ACC	PPV	NPV	TPR
VGG16	0,9991	0,9771	0,9955	0,9601	0,9586
Inception	0,9980	0,9570	0,9946	0,9247	0,9190

Tab. 3.13: Matica zámen modelu VGG16.

		Skutočnosť	
		Výstrel	Pozadie
Predikcia	Výstrel	1342	6
	Pozadie	58	1394

3.5 Rozpoznanie na základe spektrogramu

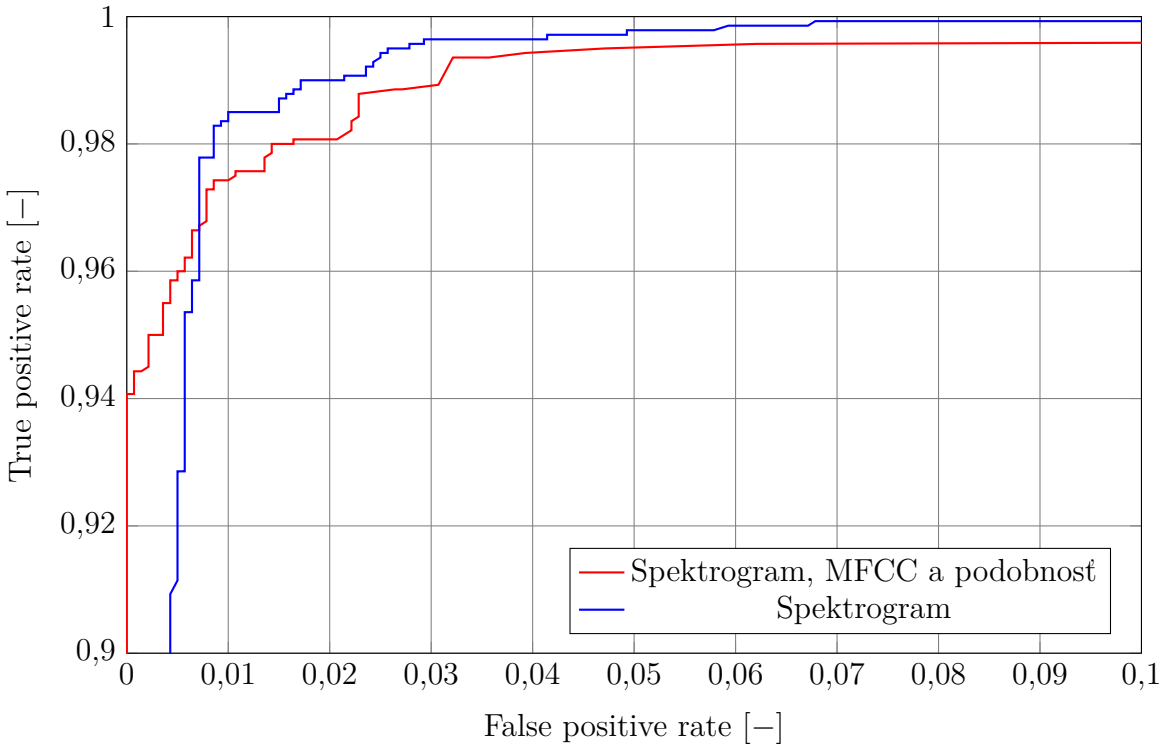
V tejto časti sa práca zaoberá porovnaním presnosti predikcie na základe spektrogramu oproti konceptu s kombináciou spektrogramu, MFCC a matice podobnosti. Použitý konvolučný model VGG16 vyžaduje na vstupe tri farebné kanály. V prípade samotného spektrogramu sú preto ostatné dva kanály nevyužité. Porovnanie spektrogramov výstrelov s rôznym odstupom SNR je na obrázku 3.12.



(a) SNR=0 dB (b) SNR=10 dB (c) Čistý výstrel

Obr. 3.12: Porovnanie spektrogramov výstrelov s rôznym odstupom signálu od šumu.

Na základe predošlých výsledkov bol pre tento test zvolený konvolučný model VGG16. Neurónová sieť bola natrénovaná na výstrely s rôznym odstupom signálu od šumu. Porovnanie kriviek ROC je zobrazené na následnom priebehu 3.13.



Obr. 3.13: Porovnanie úspešnosti rozpoznania na základne spektrogramu a kombinácie.

Rozdiel medzi krivkami konceptov je viditeľný najmä v porovnaní miesta poklesu. Modrá krivka samotného spektrogramu začína klesať skôr ako červená, čo znamená pokles presnosti rozpoznania pozitívneho prvku pri nízkom počte falošne pozitívnych predikcií. Kombináciou troch vizualizácií je podľa testu možné dosiahnuť menší podiel falošných výstrelov pri zachovanej presnosti rozpoznania skutočných výstrelov. Nad hranicou $FPR = 0,01$ dosahuje spektrogram mierne vyššiu presnosť čo znamená, že v prípade ak nieje vyžadovaný podiel falošných predikcií menší ako 1 % je vhodnejšie zvoliť vizualizáciu spektrogramom. Vyčíslenie úspešnosti rozpoznania v tabuľke 3.14 a matica zámen 3.17 boli počítané pre rovnakú chybovosť tak, aby falošných predikcií výstrelov bolo maximálne 0,5 % z celkovej testovacej množiny 1400 obrázkov.

Tab. 3.14: Vyčíslenie úspešnosti rozpoznania na základe spektrogramu a kombinácie.

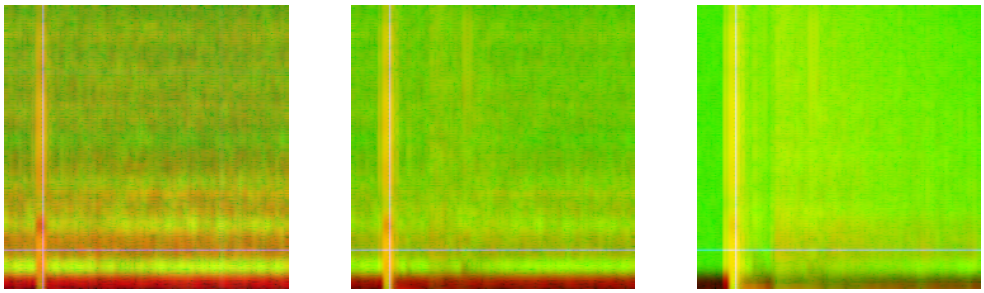
Testovaný model	AUC	ACC	PPV	NPV	TPR
Spektrogram, MFCC, podobnosť	0,9991	0,9771	0,9955	0,9601	0,9586
Samotný spektrogram	0,9976	0,9525	0,9953	0,9165	0,9093

Tab. 3.15: Matica zámen modelu trénovaného na samotné spektrogramy.

		Skutočnosť	
		Výstrel	Pozadie
Predikcia	Výstrel	1273	6
	Pozadie	127	1394

3.6 Rozpoznanie na základe podvzorkovaného signálu

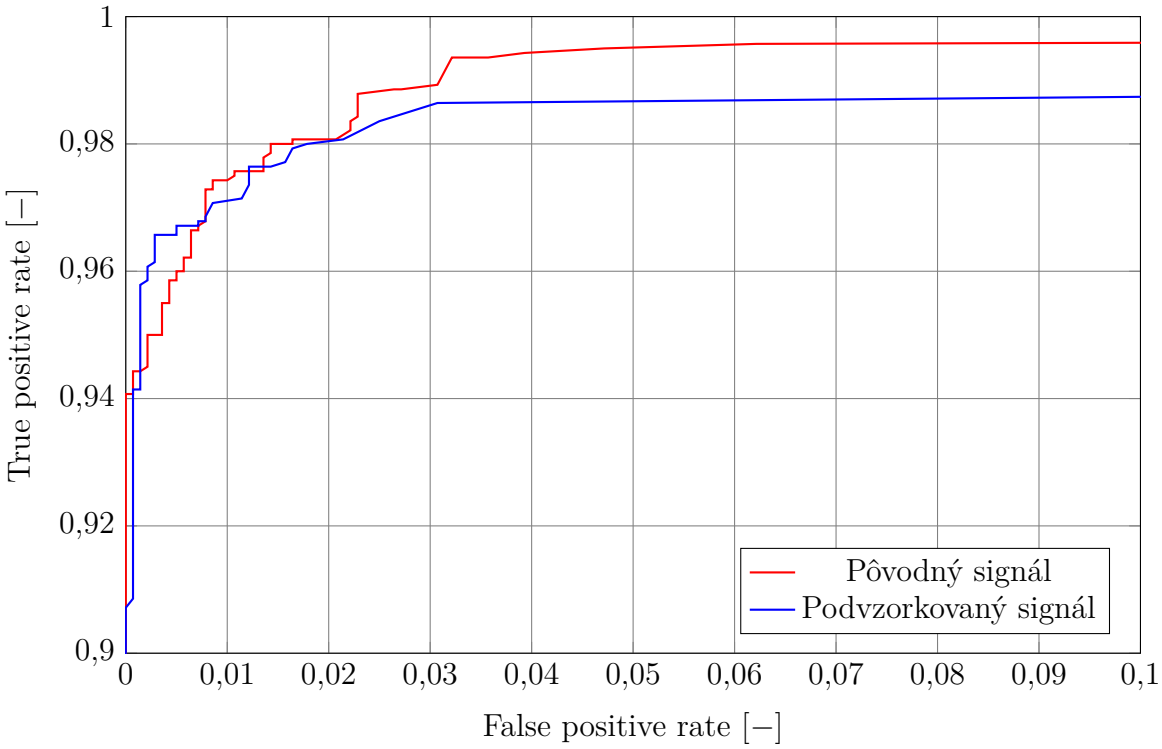
V tejto časti je snahou zistiť vplyv podvzorkovania signálu na výslednú presnosť predikcie. Pred výpočtom spektra, MFCC a podobnosti bol signál podvzorkovaný na vzorkovaciu frekvenciu 8 kHz. Výsledné vizualizácie na obrázku 3.14 sú výstrely s rôznym odstupom od ruchu pozadia.



(a) SNR=0 dB (b) SNR=10 dB (c) Čistý výstrel

Obr. 3.14: Porovnanie vizualizácií podvzorkovaných výstrelov s rôznym odstupom signálu od šumu.

Na porovnaní ROC kriviek 3.15 vidno, že priebehy sa takmer zhodujú. Veľkú podobnosť úspešnosti vidno taktiež na vyčíslených hodnotách v tabuľke 3.16. Podvzorkovanie ovplyvní presnosť len minimálne, vďaka čomu je možné dosiahnuť rovnakú presnosť predikcie pri 5-násobnom znížení dátového toku. Práve to je dôležité pri návrhu aplikácie, ktorá by mala predikovať v reálnom čase. Po prevzorkovaní je signál frekvenčne obmedzený a neobsahuje vysokofrekvenčné zložky nad hranicou 4 kHz. Je teda možné predpokladať, že pre rozpoznanie boli tieto zložky menej podstatné. Toto zistenie však nemusí platiť v prípade rozpoznávania iných zvukových udalostí ako výstrelov.



Obr. 3.15: Porovnanie úspešnosti rozpoznania na základne pôvodného a podvzorkovaného signálu.

Tab. 3.16: Vyčíslenie úspešnosti rozpoznania na základne pôvodného a podvzorkovaného signálu.

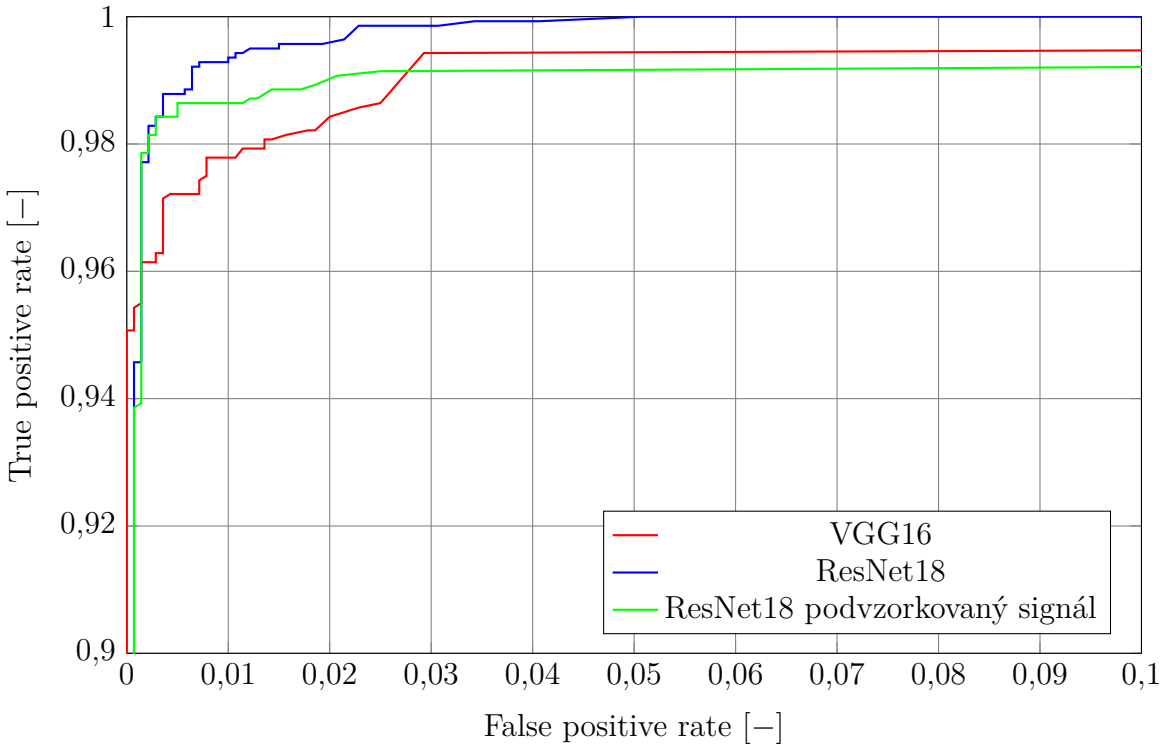
Testovaný model	AUC	ACC	PPV	NPV	TPR
Pôvodný signál	0,9991	0,9771	0,9955	0,9601	0,9586
Podvzorkovaný signál	0,9992	0,9739	0,9985	0,9517	0,9493

Tab. 3.17: Matica zámen modelu trénovaného na podvzorkovaný signál.

		Skutočnosť	
		Výstrel	Pozadie
Predikcia	Výstrel	1329	2
	Pozadie	71	1398

3.7 Trénovanie parametrov konvolučného modelu

V predchádzajúcich prípadoch boli parametre konvolučnej vrstvy uzamknuté a trénovaním sa nastavovali len váhy plne prepojenej a výstupnej vrstvy. V tejto časti je použitý model ResNet18. Trénovaním boli nastavené parametre všetkých vrstiev siete, vrátane konvolučných. Celkový počet trénovateľných parametrov modelu ResNet18 je 11 186 626. Výsledky sú porovnané s modelom VGG16 s pretrénovanými váhami konvolučných vrstiev v grafe 3.16. Medzi porovnaniami je taktiež výsledok testu s podvzorkovaním signálu.



Obr. 3.16: Porovnanie úspešnosti modelov ResNet18 a VGG16.

Vyčíslenia presnosti odpovedajú rovnako nastavenej chybovosti falošne pozitívnych predikcií 0,5 %. Model VGG16 dosahuje v tomto porovnaní nižšiu presnosť, čo môže byť spôsobené tým, že obrazová databáza ImageNet na ktorej bol model trénovaný neobsahuje vizualizácie zvuku použité v tejto práci. Použitím trénovaného modelu ResNet18 je možné naučiť sieť, ktoré artefakty vo vizualizáciách zvuku sú pre rozpoznanie dôležité, čím sa zvyšuje celková presnosť rozpoznania. Oblasti obrazu, kde nastal najväčší počet aktivácií neurónových prepojení je možné zistiť pomocou mapy aktivácií popísanej v časti 3.7.1. Podvzorkovanie signálu spôsobilo mierne väčší rozdiel úspešnosti ako v prípade predošlého testu, no k výraznému poklesu nedošlo. Pri nízkych hodnotách FPR je presnosť takmer rovnaká.

Tab. 3.18: Vyčíslenie úspešnosti rozpoznania pri použití modelu ResNet18.

Testovaný model	AUC	ACC	PPV	NPV	TPR
VGG16	0,9991	0,9771	0,9955	0,9601	0,9586
ResNet18	0,9997	0,9914	0,9950	0,9879	0,9879
ResNet18 podvzorkovaný signál	0,9990	0,9907	0,9971	0,9845	0,9843

Tab. 3.19: Matica zámen modelu ResNet18.

		Skutočnosť	
		Výstrel	Pozadie
Predikcia	Výstrel	1383	7
	Pozadie	17	1393

Tab. 3.20: Matica zámen modelu ResNet18 s podvzorkovaním signálu.

		Skutočnosť	
		Výstrel	Pozadie
Predikcia	Výstrel	1378	4
	Pozadie	22	1396

3.7.1 Mapa aktivácií

V každom kroku spracovania obrazu v konvolučných modeloch môžeme zobraziť výstup filtrácie ako mapu príznakov. Namiesto pravdepodobnosti zaradenia do jednotlivých tried môžeme zobraziť mapu aktivácií, ktorá zvýrazňuje oblasti obrazu použité sieťou na identifikáciu triedy. Podľa mapy je teda možné približne lokalizovať objekt a určiť ktoré regióny boli pre sieť relevantné. Pre výpočet mapy aktivácií

M_c je potrebné poznať váhové vektory w_k^c, ktorými je násobená k-tá mapa príznakov $f_k(x, y)$ v konvolučnej vrstve. Postup výpočtu je naznačený vzťahom 3.3, podľa [16].

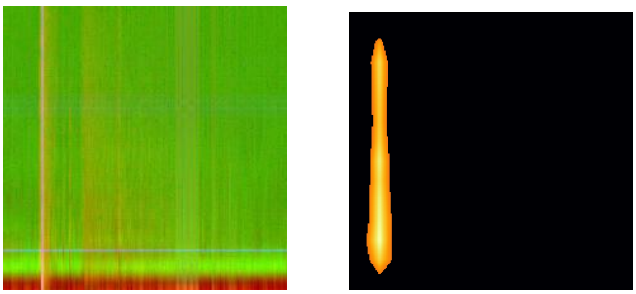
$$M_c(x, y) = \sum_k w_k^c f_k(x, y) \tag{3.3}$$

Príklad máp aktivácií pri spracovaní vizualizácie výstrelu je zobrazený na obrázku 3.17. Mapy boli získané z výstupu druhej, štvrtej a šiestej konvolučnej vrstvy modelu ResNet18. Celková architektúra modelu je zobrazená v tabuľke 3.21.

Tab. 3.21: Architektúra modelu ResNet18 [12].

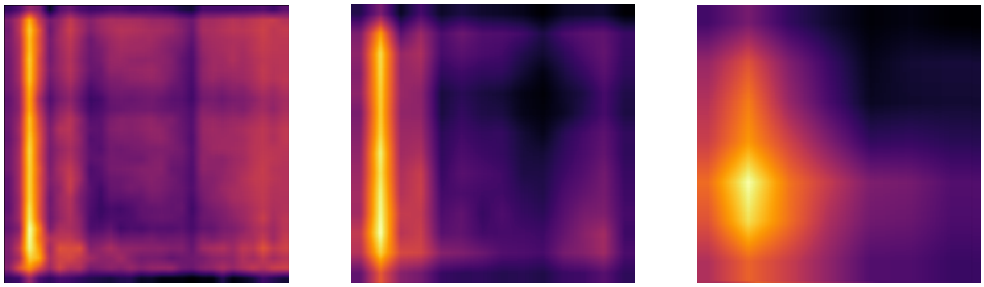
Názov vrstvy	Výstupné rozlíšenie	Bloky
conv1	112×112	7×7 max pool, stride 2
conv2_x	56×56	3×3 max pool, stride 2 $\begin{bmatrix} 3\times3,\ 64 \\ 3\times3,\ 64 \end{bmatrix} \times 2$
conv3_x	28×28	$\begin{bmatrix} 3\times3,\ 128 \\ 3\times3,\ 128 \end{bmatrix} \times 2$
conv4_x	14×14	$\begin{bmatrix} 3\times3,\ 256 \\ 3\times3,\ 256 \end{bmatrix} \times 2$
conv5_x	7×7	$\begin{bmatrix} 3\times3,\ 512 \\ 3\times3,\ 512 \end{bmatrix} \times 2$
	1×1	average pool, 1000-d fc, softmax

V nižších vrstvách modelu je postupne znižované rozlíšenie a to na 14×14 v štvrtej a 7×7 v šiestej vrstve. Pred zobrazením boli mapy preto zväčšené na rovnaké rozlíšenie ako vstupný obraz 244×244.



(a) Vstupný obraz (b) Mapa aktivácií

Obr. 3.18: Prahovaná mapa aktivácií v štvrtej vrstve a vstupná vizualizácia výstrelu.



(a) Druhá vrstva (b) Štvrtá vrstva (c) Šiesta vrstva

Obr. 3.17: Mapy aktivácií modelu ResNet18 vo vybraných vrstvách pri spracovaní vizualizácie výstrelu.

Rozdiel v koncentráciách aktivácií medzi vizualizáciou výstrelu a náhodného pozadia je jednoznačne viditeľný z následného porovnania. Na obrázku 3.19 vidno porovnanie máp aktivácií vizualizácií nahrávok rôzneho charakteru. Najväčšie aktivácie sú sústredené v rozdielnych častiach obrazu ako v prípade výstrelov.

(a) Detský krik (b) Stavebné náradie (c) Hudba

Obr. 3.19: Porovnanie máp aktivácií zvukových pozadí rôzneho charakteru.

4 VÝSLEDNÁ APLIKÁCIA

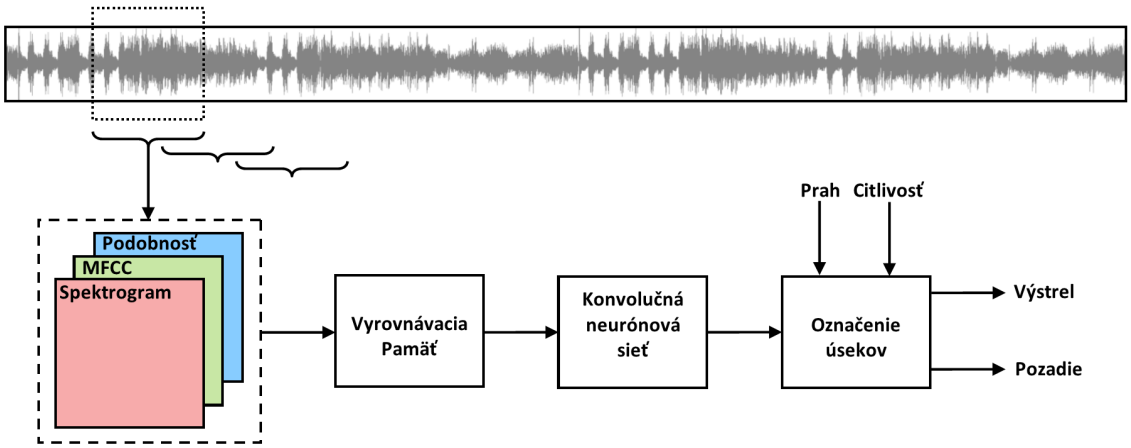
4.1 Analýza zvukovej nahrávky

4.1.1 PyInstaller

Pre spustenie funkčného Python skriptu je potrebné nainštalovať na danom zariadení všetky balíky a nástroje použité v aplikácií. Pomocou nástroja PyInstaller je však možné zaobaliť skript spolu so všetkými potrebnými súčasťami do jednej spustiteľnej aplikácie. Takto zabalená aplikácia je potom prenositeľná na viaceré zariadenia bez nutnosti ďalších inštalácií.

4.1.2 Návrh funkčnej časti

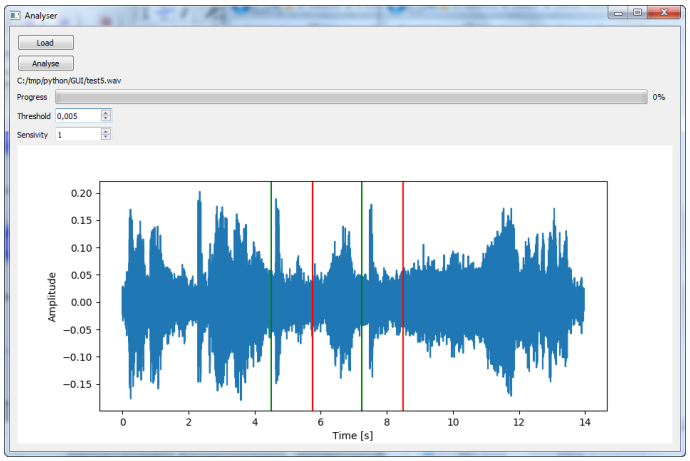
Celkový návrh výslednej aplikácie pozostával z návrhu funkčnej časti v jazyku Python a grafického rozhrania. Vstupom aplikácie je ľubovoľná zvuková nahrávka, v ktorej je očakávaný výstrel. Proces spracovania signálu v aplikácií musí byť rovnaký, ako pri generovaní trénovacej množiny. Preto bolo základnou úlohou spracovať nahrávku v jazyku Python rovnako, ako v programe MATLAB pri tvorbe obrázkovej databázy pre trénovanie. Stereo nahrávky sú spracované ako mono. Dĺžka okna spracovania je jedna sekunda so štvrtinovým posunom. Z týchto sekundových intervalov je počítaný spektrogram, MFCC koeficienty a matica podobnosti. Po dokončení generovania vizualizácií je načítaný natrénovaný model neurónovej siete. Spracovaných je 10 vizualizácií paralelne. Grafické znázornenie funkčnosti aplikácie je zobrazené na obrázku 4.1.



Obr. 4.1: Diagram aplikácie.

4.1.3 Grafické rozhranie

Jednoduché grafické prostredie aplikácie bolo navrhnuté na platforme PyQt5. Hlavné okno aplikácie je zobrazené na obrázku 4.2 a je v ňom možné vybrať analyzovanú nahrávku, sledovať priebeh analýzy a zmeniť citlivosť detekcie. Citlivosť 1 znamená, že na detekciu výstrelu postačuje aby bol pozitívny jeden zo štyroch prípadov v rámci spracovania sekundového intervalu s prekrytím.



Obr. 4.2: Hlavné okno aplikácie.

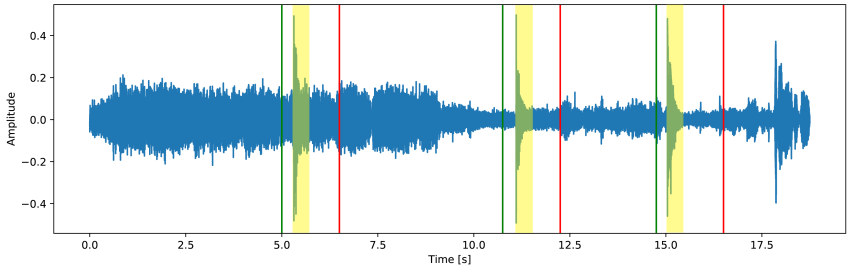
Výber nahrávky je obmedzený na zvukové súbory vo formáte WAV a je možné zvoliť nahrávku s ľubovoľnou dĺžkou trvania. Počas analýzy aplikácia zobrazuje aktuálny stav a vývoj. Po dokončení je zobrazený časový priebeh nahrávky s vyznačenými úsekmi zvuku výstrelu. V tomto momente je možné meniť citlivosť a prah rozhodovania detektoru. Zmena sa prejaví ihneď v okne s priebehom analyzovanej nahrávky. Vďaka tomu je možné porovnať výsledky detekcie pri rôznom nastavení.

4.1.4 Výsledky analýzy nahrávok

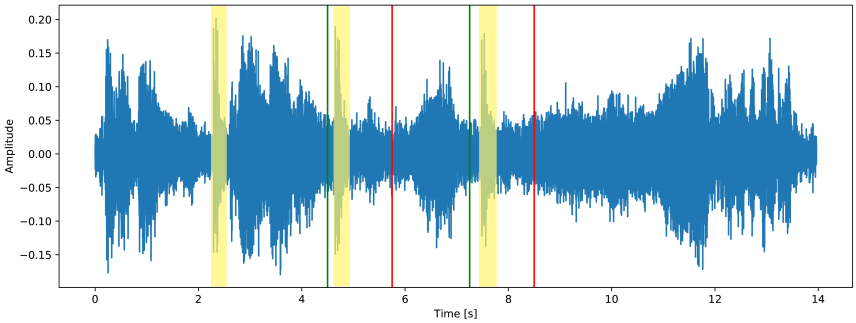
Pomocou aplikácie boli analyzované dlhšie trvajúce nahrávky obsahujúce výstrely. Zvuky reálnych strelných zbraní boli nahrané na strelnici s použitím smerového mikrofónu. Tieto zvuky boli primiešané k ruchu rôzneho charakteru získaného z filmových úryvkov. Následné obrázky zobrazujú časové priebehy zvukových signálov s označenými úsekmi výstrelov. Zelená a červená čiara ohraničujú miesta, ktoré našla aplikácia a žltou farbou sú zvýraznené časti kde skutočne došlo k výstrelu.

Na obrázku 4.3 vidno, že aplikácia pri nastavení prahu 0,01 a citlivosti 1 správne označila všetky tri úseky obsahujúce výstrel a nepomýlila ju zrážka dvoch áut v rušnom pozadí. Ďalší príklad je obrázok 4.4, kde aplikácia nedokázala rozpoznať výstrel na začiatku. V tejto nahrávke sú výstrely viac prekryté ruchom pozadia, no

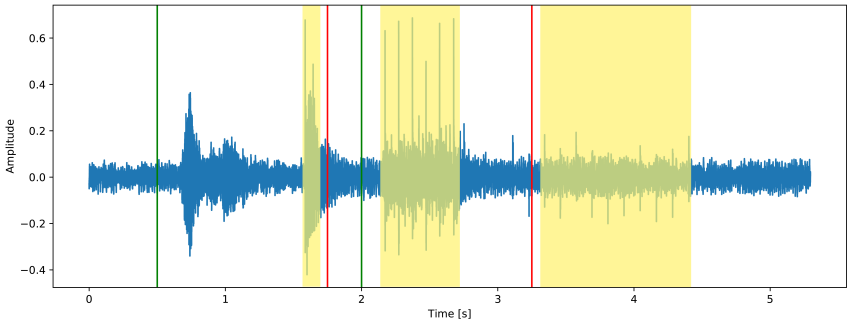
subjektívne sú stále rozpoznateľné. Prah 0,002 a citlivosť 1 v tomto prípade boli nastavené na veľmi nízke hodnoty. Ako príklad nesprávnej analýzy slúži obrázok 4.5, kedy aplikácia s nastaveným prahom 0,94 a citlivosťou 1 označila rozbitie skla ako výstrel. Streľba zo samopalu na konci nahrávky bola subjektívne veľmi ťažko rozpoznateľná a aplikácia si s ňou taktiež neporadila.



Obr. 4.3: Analýza nahrávky *testtrack1*, ktorú aplikácia označila správne.



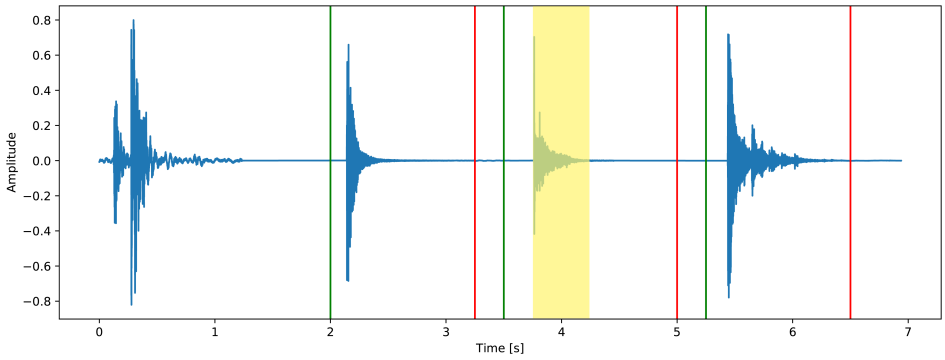
Obr. 4.4: Analýza nahrávky *testtrack2*, kde aplikácia prehliadla jeden výstrel.



Obr. 4.5: Analýza nahrávky *testtrack3*, ktorú aplikácia označila nesprávne.

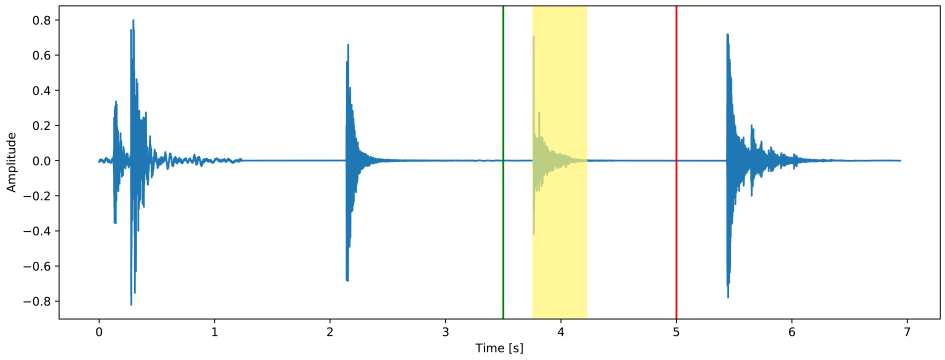
4.1.5 Vplyv citlivosti na úspešnosť rozpoznania

Predošlé testy porovnali úspešnosť rozpoznania výstrelu v rôzne zašumených prostrediach a ukázali vplyv rušnosti pozadia na úspešnosť. V následovnom teste bola analyzovaná nahrávka obsahujúca zvukové udalosti s podobným charakterom, konkrétne zabuchnutie dverí, úder na bubon, rozbitie skla a hľadaný výstrel. V prvom prípade nahrávka neobsahovala žiadne ruchové pozadie, no kvôli podobnosti zvukov sa napriek tomu jedná o náročnú úlohu. Prah rozpoznania bol nastavený 0,94 podľa ROC analýzy použitého modelu tak, aby len 0,1 % prípadov spadalo medzi falošne pozitívne predikcie.



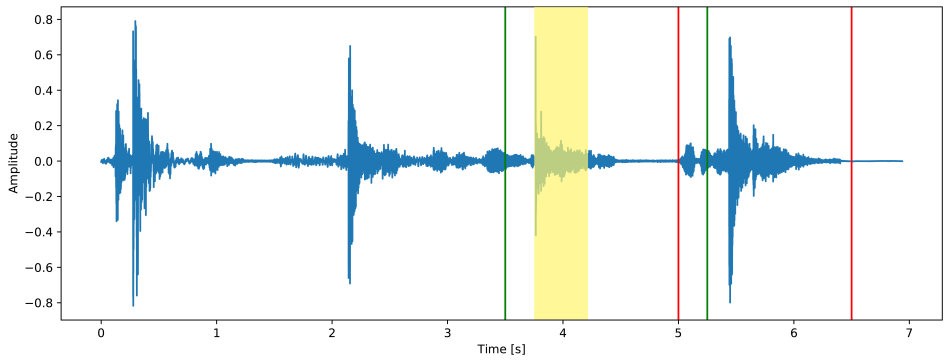
Obr. 4.6: Nesprávne označená čistá nahrávka *clear*.

Na obrázku 4.6 vidno ako aplikácia nesprávne označila úder na bubon a rozbitie skla napriek nastaveniu veľmi prísneho prahu. Označené úseky po zmene citlivosti z 1 na 2 sú zobrazené na následovnom obrázku 4.7.

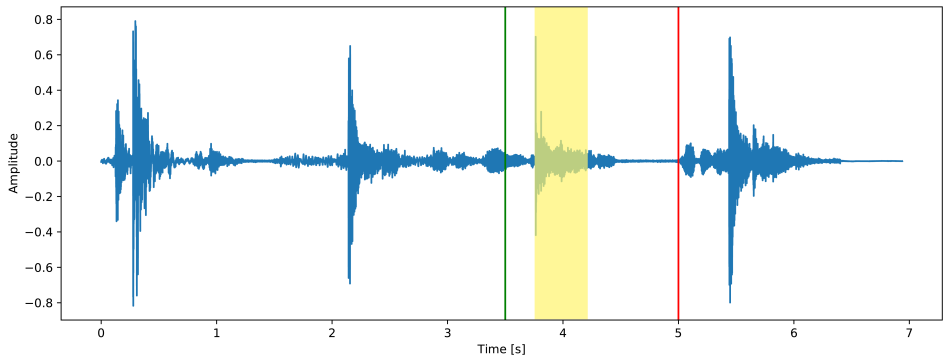


Obr. 4.7: Správne označená čistá nahrávka *clear*.

Zníženie citlivosti v zmysle počtu za sebou nasledujúcich predikcií spôsobilo, že aplikácia označila správne úsek výstrelu. Rovnaký postup bol zopakovaný so zarušenou nahrávkou s pridanou rečou v pozadí.



Obr. 4.8: Nesprávne označená zarušená nahrávka *noisy*.



Obr. 4.9: Správne označená zarušená nahrávka *noisy*.

Výsledok je veľmi podobný, no v tomto prípade bol prah rozpoznania nastavený na podstatne nižšiu hodnotu 0,012 ako v predošlom teste. Tento experiment teda ukázal, že skutočná úspešnosť rozpoznania výstrelu v reálnom prostredí je vo veľkej miere závislá na nastavenom prahu a citlivosti. Na správne nastavenie týchto parametrov vplýva najmä rušnosť prostredia a teda odstup signálu od šumu. V prostredí s vyššou úrovňou šumu je pre dosiahnutie vyššej úspešnosti potrebné znížiť prah, čo však môže viesť k nárastu počtu falošne pozitívnych predikcií. Preto v prípade zavedenia postupu do reálnej aplikácie je vhodné algoritmus doplniť o detektor zvukovej aktivity a analyzovať len vybrané úseky.

4.2 Analýza v reálnom čase

4.2.1 Optimalizácia spracovania signálu

Pred návrhom aplikácie pracujúcej v reálnom čase bolo potrebné znížiť čas extrakcie príznakov. To bolo prevedené optimalizáciou častí programu súvisiacich s výpočtom spektrogramu, MFCC a matice podobnosti. Opakovacie slučky, pri ktorých to bolo možné, boli nahradené vektorovými operáciami, ktoré spotrebujú výrazne menej výpočtového času. Premenné, ktoré nemenia hodnotu počas spracovania boli načítané a uložené do dočasnej pamäte pri spustení programu. Zároveň bol znížený počet vzorkov furiérovej transformácie pri výpočte spektrogramu na 2048. Po týchto úpravách programu bol čas spracovania sekundového intervalu zvukového signálu približne 80 ms. K celkovému času spracovania je však potrebné pripočítať čas predikcie konvolučnou neurónovou sieťou.

4.2.2 Porovnanie časov spracovania

V časti 3.7 je popísaný test s reziduálnym konvolučným modelom ResNet18. Tento experiment bol motivovaný hľadaním rýchlejšej alternatívy oproti modelu VGG16. Veľkým prínosom bolo zistenie, že model ResNet18 pretrénovaný vo všetkých vrstvách nedokazoval len vyššiu úspešnosť rozpoznania ale tiež podstatne nižší čas predikcie. Porovnanie časov spracovania je v tabuľke 4.1. Výsledná aplikácia pracujúca v reálnom čase bola preto postavená na tomto modely a zároveň bol využitý poznatok, že podvzorkovanie výrazným spôsobom nezhorší presnosť rozpoznania výstrelu.

Tab. 4.1: Porovnanie časov spracovania sekundového intervalu signálu.

Extrakcia príznakov	80 ms
Predikcia VGG16	400 ms
Predikcia ResNet18	50 ms

Použitím ResNet18 dosahuje výsledný čas extrakcie a predikcie sekundového intervalu 130 ms. V prípade štvrtinového posunu okna spracovania je čas štvornásobný, teda 520 ms. S takto nastavenými parametrami je preto možné analyzovať signál v reálnom čase s dostatočnou časovou rezervou. Pri tomto meraní prebiehali výpočty na procesorových vláknach 4-jadrového CPU Intel Core i7 s ôsmimi vláknami a s taktovacou frekvenciou 2 GHz bez použitia grafickej akcelerácie.

4.2.3 Návrh funkčnej časti

Funkčná časť je z veľkej časti prevzatá z predošlej aplikácie. Spracovanie signálu a grafické rozhranie sú spustené v dvoch vláknach, vďaka čomu je aplikácia responzívna počas prehrávania. Načítané dáta zvukového súboru sú radené do fronty po sekundových intervaloch.

```
while data:
    self.q.put(data, timeout=timeout)
    data = f.buffer_read(self.blocksize, dtype='float32')
```

Zvuková karta vyvoláva *callback* v ktorom prijíma dáta z fronty a zároveň je nad týmito dátami spustená analýza. Definícia *callbacku*, kde dochádza k volaniu vlastnej funkcie *analyse* je následovná:

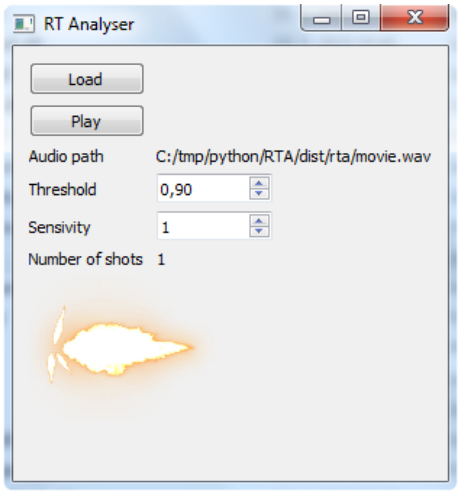
```
def callback(self,outdata, frames, time, status):
        assert frames == self.blocksize
        if status.output_underflow:
            raise sd.CallbackAbort
        assert not status
        try:
            data = self.q.get_nowait()
        except queue.Empty:
            raise sd.CallbackAbort
        if len(data) < len(outdata):
            outdata[:len(data)] = data
            d = b'\x00' * (len(outdata) - len(data))
            outdata[len(data):] = d
            self.analyse(data)
            raise sd.CallbackStop
        else:
            outdata[:] = data
            self.analyse(data)
```

Vďaka tomuto prístupu, je v prípade potreby možné ľahko prepojiť *stream* zo súboru na zvukový vstup karty a analyzovať signál z mikrofónu. Po vizualizácií časti signálu a úprave vstupných dát sa vykoná samotná predikcia príkazom *predict_generator*.

```
prob = self.model.predict_generator(test_generator,
                                    steps=1)
```

4.2.4 Grafické rozhranie

Grafické menu zobrazené na obrázku 4.10 je podobné ako v prípade predošlej aplikácie. Namiesto časového priebehu je však zobrazený obrázok plameňa v momente, kedy pri prehrávaní zaznie výstrel. Súčasne je zobrazené počítadlo výstrelov. Citlivosť a prah rozhodovania je možné meniť počas prehrávania nahrávky. Prah opäť znamená hodnotu pravdepodobnosti, ktorá musí byť prekročená aby bol úsek detekovaný ako pozitívny. Citlivosť určuje, koľko za sebou nasledujúcich úsekov musí byť pozitívnych aby bol detekovaný výstrel.



Obr. 4.10: Grafické prostredie aplikácie pre analýzu v reálnom čase.

5 ZÁVER

Výsledky práce potvrdzujú, že konvolučné modely určené na rozpoznávanie objektov v obraze je možné s vysokou úspešnosťou použiť taktiež na rozpoznávanie zvukových udalostí. Trénovacie a testovacie množiny nahrávok je potrebné čo najviac priblížiť k reálnym podmienkam, čo bolo v práci realizované pridávaním náhodného časového posunu a šumového pozadia. Hlboká neurónová sieť vykazuje najlepšie výsledky v prípade, že trénovacia množina obsahuje nahrávky s rôznymi odstupmi signálu od šumu. V prípade testovania množiny s nízkym odstupom od šumu, kedy bol výstrel subjektívne ťažko rozpoznateľný, vyhodnotil algoritmus viac ako polovicu výstrelov správne. Takto natrénovaný model teda dosahuje dobré výsledky pri testovaní výstrelmi so šumovým pozadím pri zachovanej vysokej úspešnosti rozpoznania čistého výstrelu.

Konvolučné modely VGG16 a InceptionV3 s predtrénovanými váhami na obrazovej databáze ImageNet sú použiteľné na klasifikáciu zvukových udalostí napriek tomu, že ImageNet neobsahuje žiadne vizualizácie zvuku. Pokusy dokazujú, že vyššiu úspešnosť rozpoznania je možné dosiahnuť pretrénovaním parametrov konvolučných vrstiev. Tento prístup však môže v prípade rozsiahlej architektúry siete zabrať podstatne vyšší čas, keďže stúpa počet trénovateľných parametrov.

V prípade návrhu aplikácie pre analýzu zvuku v reálnom čase sú podstatné časy extrakcie príznakov a predikcie. Čas extrakcie príznakov závisí od veľkosti dátového toku zvuku a množstva extrahovaných príznakov. Porovnaním úspešnosti predikcie na základe samotného spektrogramu a kombinácie spektrogramu, MFCC a matice podobnosti bolo zistené, že samotný spektrogram môže dokonca dosahovať vyššiu presnosť, ktorá však výrazne klesne pri požiadavke nízkej chybovosti falošne pozitívnych výstrelov. V tom prípade je vhodnejšie použiť testovanú kombináciu.

Experiment ukázal, že zvýšenie rýchlosti spracovania pri zachovanej úspešnosti je možné dosiahnuť podvzorkovaním signálu. Vplyv 5-násobného zníženia dátového toku podvzorkovaním na úspešnosť rozpoznania bol minimálny. Čas predikcie závisí najmä od počtu operácií, ktoré musí model vykonať pri spracovaní jedného snímku. Ako najvhodnejší z tohto hľadiska bol vybraný model ResNet18, ktorý nedosahoval len nižší čas spracovania ale po pretrénovaní všetkých parametrov taktiež najvyššiu presnosť.

Na základe týchto zistení boli navrhnuté dve aplikácie, ktoré využívajú pretrénovaný model ResNet18 a pracujú s podvzorkovaným signálom. Prvá aplikácia analyzuje zvukovú nahrávku a označuje miesta, kde pravdepodobnosť výskytu výstrelu prekročila požadovanú hranicu. Taktiež umožňuje obmedziť skutočnú detekciu len na prípady, kedy dôjde k niekoľkým následujúcim detekciám pri analýze s prekrytím. Druhá aplikácia analyzuje zvukovú nahrávku v reálnom čase počas prehrávania. Do-

siahnutý čas spracovania sekundového úseku je približne pol sekundy. Aplikácia je teda schopná pracovať na procesorových jadrách bez použitia grafickej akcelerácie.

Výstrel ako vybraná zvuková udalosť, má pomerne špecifický charakter. Niektoré zistenia tohto výskumu preto nemusia platiť pri rozpoznávaní iných vybraných udalostí. Táto práca otvára priestor pre ďalší výskum vplyvu použitých architektúr a spôsobov predspracovania signálu pri rozpoznávaní iných zvukových udalostí pomocou hlbokého učenia.

LITERATÚRA

[1] SMÉKAL, Zdenek. *Číslicové zpracování řeči*. Skriptum Ústav telekomunikací VUT v Brne, posledná aktualizácia 2010. 134 s.

[2] MASTERS, Timothy. *Signal and image processing with neural networks*. New York: J. Wiley, c1994. ISBN 0471049638.

[3] BODDAPATI, Venkatesh, PETEF, Andrej, RASMUSSON, Jim, LUNDBERG, Lars. *Classifying environmental sounds using image recognition networks*. Publikované v Procedia Computer Science, 2017. s. 2048–2056.

[4] FOOTE, Jonathan T., COOPER, Matthew L. *Media Segmentation using Self-Similarity Decomposition*. Publikové v SPIE Storage and Retrieval for Media Databases 2003, Vol. 5021, s. 167-175.

[5] MITCHELL, Tom M. *Machine Learning*. New York: McGraw-Hill, c1997. ISBN 0070428077.

[6] GOODFELLOW, Ian, BENGIO, Yoshua, COURVILLE, Aaron. *Deep Learning*. MIT Press, 2016. 785 s.

[7] SRIVASTAVA, Nitish, HINTON, Geoffrey, KRIZHEVSKY, Alex, SUTSKEVER, Ilya, SALAKHUTDINOV, Ruslan. *Dropout: A Simple Way to Prevent Neural Networks from Overfitting*. Department of Computer Science, University of Toronto 2014. 30 s.

[8] GONZALES, Rafael C., WOODS, Richard E. *Digital Image Processing*. New Jersey: Prentice-Hall, 2002.

[9] SIMONYAN, Karen, ZISSERMAN, Andrew. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. Visual Geometry Group, Department of Engineering Science, University of Oxford 2015. 14 s.

[10] CANZIANI, Alfredo, PASZKE, Adam, CULURCIELLEO, Eugenio. *An Analysis of Deep Neural Network Models for Practical Applications*. Weldon School of Biomedical Engineering Purdue University, 2017. 7 s.

[11] SZEGEDY, Christian, VANHOUCKE, Vincent, IOFFE, Sergey, SHLENS, Jonathon, WOJNA, Zbigniew. *Rethinking the Inception Architecture for Computer Vision*. Google Inc. and University College London, 2015. 10 s.

[12] KAIMING, He, XIANGYU, ZHANG, Shaoqing, Ren, JIAN, Sun. *Deep Residual Learning for Image Recognition*. Microsoft Research 2015. 12 s.

[13] ABADI, M., BARHAM, P., CHEN, J., CHEN, Z., DAVIS, A., DEN, J., DEVIN, M., GHEMAWAT, S., IRVING, G., ISARD, M., KUDLUR, M., LEVENBERG, J., MONGA, R., MOORE, S., MURRAY, D.G., STEINER, B., TUCKER, P.A., VASUDEVAN, V., WARDEN, P., WICKE, M., YU, Y., ZHANG, X. *TensorFlow: A system for large-scale machine learning*. Publikované v OSDI, 2016. s. 265-281. ISBN 9781931971331

[14] GULLI, Antonio, PAL, Sujit. *Deep Learning with Keras*. Birmingham: Packt Publishing, c2017. ISBN 9781787128422.

[15] FAWCETT, Tom. *An introduction to ROC analysis*. Publikované v Pattern Recognition Letters 27 2005. s. 861-874.

[16] ZHOU, Bolei, KHOSLA, Aditya, LAPEDRIZA, Agata, OLIVA, Aude, TORRALBA, Antonio. *Learning Deep Features for Discriminative Localization*. Computer Science and Artificial Intelligence Laboratory, MIT 2016. 10 s.

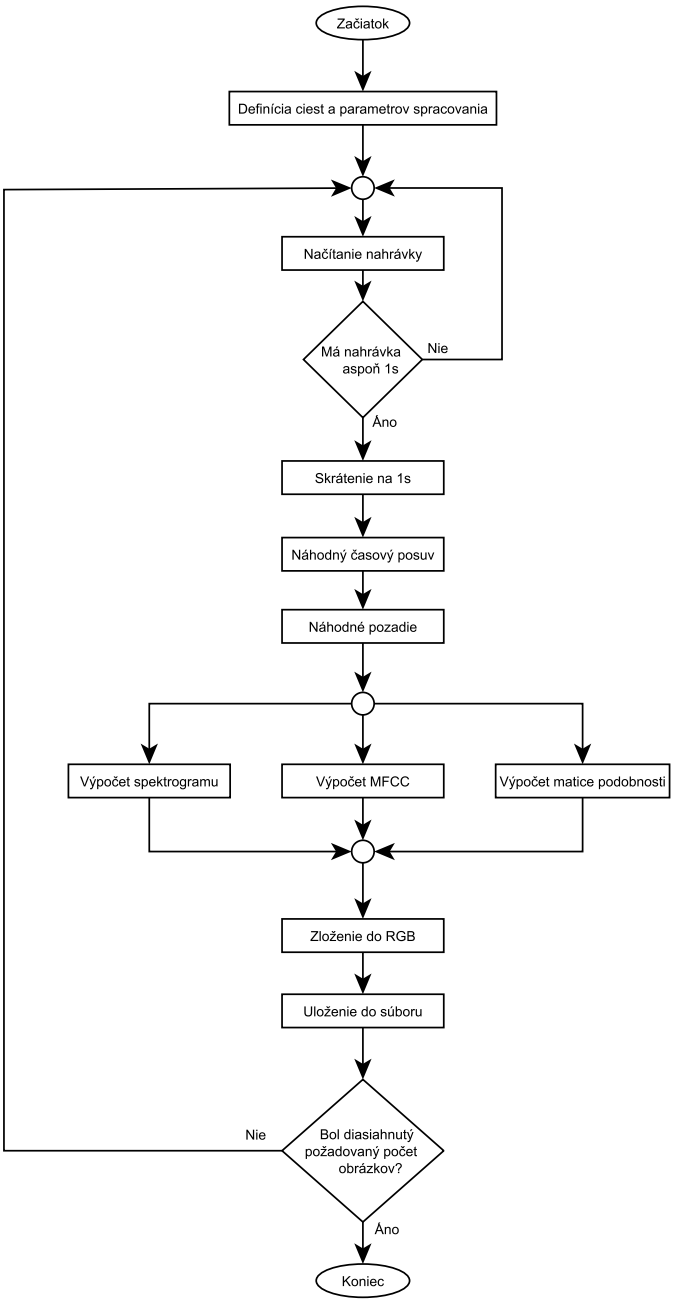
ZOZNAM SYMBOLOV, VELIČÍN A SKRATIEK

ACC	Accuracy - Presnosť
AUC	Area Under Curve - Plocha pod krivkou
API	Application Programming Interface - Aplikačné programové rozhranie
CPU	Central Processing Unit - Hlavný procesor
DFT	Discrete Fourier Transform - Diskrétna furiérova transformácia
FN	False Negative
FP	False Positive
FPR	False Positive Rate
IDFT	Inverse Discrete Fourier Transform - Inverzná diskrétna furiérova transformácia
IoT	Internet of things - Internet vecí
ln	Prirodzený logaritmus
MFCC	Mel Frequency Cepstral Coefficients - Melovské kepstrálne koeficienty
MSE	Mean Square Error - Stredná kvadratická odchýlka
NPV	Negative Predictive Value
PPV	Positive Predictive Value
RGB	Red Green Blue - Farebné vrstvy obrazu
RMS	Root Mean Square - Efektívna hodnota
ROC	Receiver operating characteristic - charakteristická krivka prijímača
SNR	Signal Noise Ratio - Odstup signálu od šumu
TN	True Negative
TP	True Positive
TNR	True Negative Rate
TPR	True Positive Rate

ZOZNAM PRÍLOH

A VÝVOJOVÉ DIAGRAMY

A.1 Spracovanie nahrávok výstrelov v programe MATLAB



B ZOZNAM PRÍLOH NA DISKU

- Analyser ..... Spustiteľná aplikácia pre analýzu nahrávky
- RealTimeAnalyser ..... Spustiteľná aplikácia pre analýzu v reálnom čase
- source ..... Zdrojové súbory
  - matlab
  - python
- tracks ..... Testovacie nahrávky
- text.pdf ..... Text záverečnej práce